

# **Отчёт по лабораторной работе №2**

**Управление версиями**

Лев Сирота НБИбд-04-22

# Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Вывод	11
4	Контрольные вопросы	12
	Список литературы	16

## Список иллюстраций

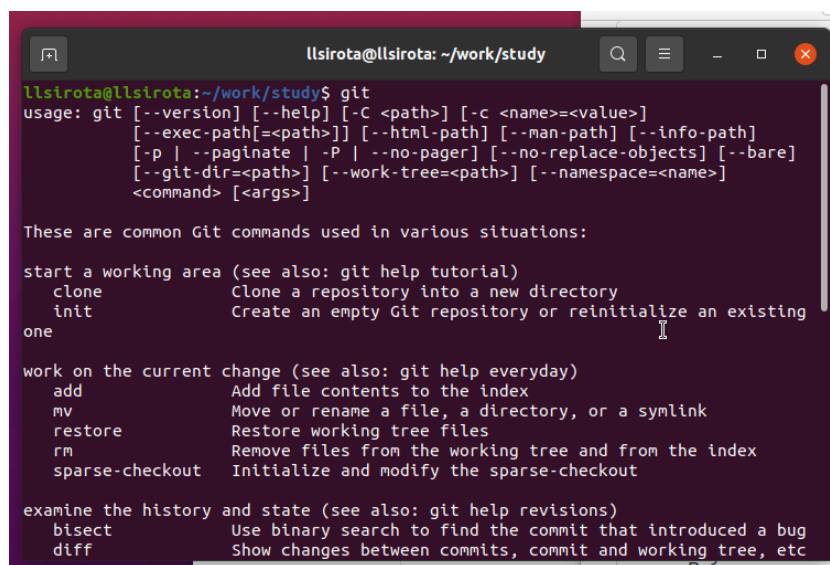
2.1	Загрузка пакетов . . . . .	5
2.2	Параметры репозитория . . . . .	6
2.3	rsa-4096 . . . . .	6
2.4	ed25519 . . . . .	7
2.5	GPG ключ . . . . .	7
2.6	GPG ключ . . . . .	8
2.7	Параметры репозитория . . . . .	8
2.8	Связь репозитория с аккаунтом . . . . .	9
2.9	Загрузка шаблона . . . . .	9
2.10	Первый коммит . . . . .	10

# 1 Цель работы

Целью данной работы является изучение идеологии и применения средств контроля версий и освоение умений работать с git.

## 2 Выполнение лабораторной работы

Устанавливаем git, git-flow и gh.

A terminal window titled 'llesirota@llesirota: ~/work/study' showing the output of the 'git' command. The output includes the usage of git, a list of common Git commands, and their descriptions.

```
llesirota@llesirota:~/work/study$ git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
        [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
        [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
        [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
        <command> [<args>]

These are common Git commands used in various situations:


start a working area (see also: git help tutorial)
    clone      Clone a repository into a new directory
    init       Create an empty Git repository or reinitialize an existing
one

work on the current change (see also: git help everyday)
    add        Add file contents to the index
    mv         Move or rename a file, a directory, or a symlink
    restore    Restore working tree files
    rm         Remove files from the working tree and from the index
    sparse-checkout  Initialize and modify the sparse-checkout

examine the history and state (see also: git help revisions)
    bisect     Use binary search to find the commit that introduced a bug
    diff       Show changes between commits, commit and working tree, etc
```

Рис. 2.1: Загрузка пакетов

Зададим имя и email владельца репозитория, кодировку и прочие параметры.

```
llsirota@llsirota: ~/work/study
reset          Reset current HEAD to the specified state
switch         Switch branches
tag            Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
fetch          Download objects and refs from another repository
pull           Fetch from and integrate with another repository or a local
branch         branch
push           Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
llsirota@llsirota:~/work/study$
llsirota@llsirota:~/work/study$
llsirota@llsirota:~/work/study$ git config --global user.name "llsirota0"
llsirota@llsirota:~/work/study$ git config --global user.email "1132221809@pfur.ru"
llsirota@llsirota:~/work/study$ git config --global core.quotePath false
llsirota@llsirota:~/work/study$ git config --global init.defaultBranch master
llsirota@llsirota:~/work/study$ git config --global core.autocrlf input
llsirota@llsirota:~/work/study$ git config --global core.safecrlf warn
llsirota@llsirota:~/work/study$
```

Рис. 2.2: Параметры репозитория

Создаем SSH ключи

```
llsirota@llsirota:~/work/study$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/llsirota/.ssh/id_rsa):
/home/llsirota/.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/llsirota/.ssh/id_rsa
Your public key has been saved in /home/llsirota/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:uLtqzAhc8B2kgiC2QpkPXBWXSJe5gGJrhCQUCGxSTJU llsirota@llsirota
The key's randomart image is:
+---[RSA 4096]-----+
|@@0.+==oo .o      |
|XB==E.o.+o        |
|=0==+o. o.o.       |
|o o+   ...        |
|...   .S          |
|. + .             |
|. = .             |
|. . .             |
|. . .             |
|. o.              |
+---[SHA256]-----+
llsirota@llsirota:~/work/study$
```

Рис. 2.3: rsa-4096

```
llsirota@llsirota: ~/work/study
+----[SHA256]-----+
llsirota@llsirota:~/work/study$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/llsirota/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/llsirota/.ssh/id_ed25519
Your public key has been saved in /home/llsirota/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:rEF1M1hfk6W+YM1rINMdr3uHMSXLPsvv52LDvLV2K+4 llsirota@llsirota
The key's randomart image is:
+---[ED25519 256]---+
|      .o=  oo.|
|    ... + ..o |
|      .  .o  |
|    .  .  .o.|
|    .  .  =.o.|
|    .  S o  =.=+|
|      o  + o==+|
|      +==.|
|      .  .oXo+|
|      .E*+*|
+----[SHA256]-----+
llsirota@llsirota:~/work/study$
```

Рис. 2.4: ed25519

Создаем GPG ключ

```
llsirota@llsirota: ~/work/study
"llsirota0 <1132221809@pfur.ru>"
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /home/llsirota/.gnupg/trustdb.gpg: trustdb created
gpg: key 9977424606D4B5FB marked as ultimately trusted
gpg: directory '/home/llsirota/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/llsirota/.gnupg/openpgp-revocs.d/35
94951EF0449C62EB798C709977424606D4B5FB.rev'
public and secret key created and signed.

pub   rsa4096 2023-02-14 [SC]
      3594951EF0449C62EB798C709977424606D4B5FB
uid           llsirota0 <1132221809@pfur.ru>
sub   rsa4096 2023-02-14 [E]

llsirota@llsirota:~/work/study$
```

Рис. 2.5: GPG ключ

Добавляем GPG ключ в аккаунт

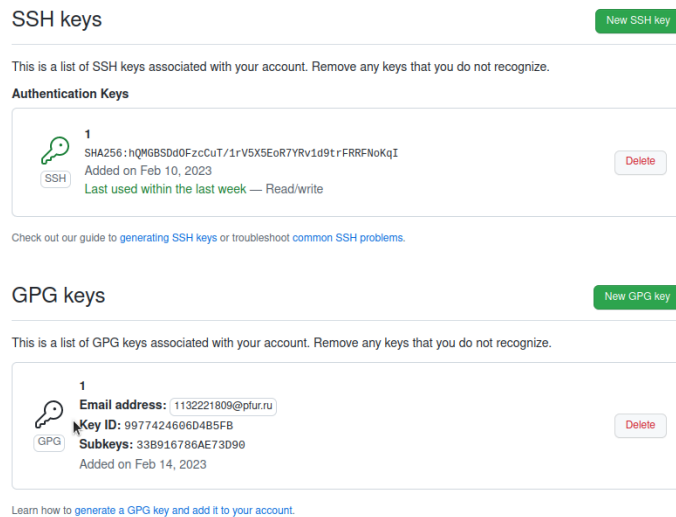


Рис. 2.6: GPG ключ

## Настройка автоматических подписей коммитов git

```
llsirota@llsirota: ~/work/study
5d95YE6m1n5jFgy14qEZSpPdbyslJdCj1JR0zX1ninbpgTE443hjMye6+Vh7YHLC
tQBwaAG+0dYCxgrVagglfL3E3Bo07yHh8ntFWGmRDVpb+koVDv7FAx9psZDVzWxd
viYBLPSe0dtCCAbX+MXkoD0ix60AztokbF0Ez4Sjo2R7eCnJUmtvfTTYy7J4ezJW
nmxBzf0SZLqBG9Vi1L9/16A65y/2oJDHPgsT3HpFw0/LLjAzHeJr5Ljqf+srw4q5
UotZR+FTDLfbptUt2aSilvDSLWT0JmXQD2STs26FeFqq6W0XKZe6UYHHSdigLEe
Qy9KQu8KwLM+9/HLU4cMGz+FM4IoJ0Qoyn7PR54jj1UIzlniqzoxJK4rAh9ScJbY
uQ0/Ae00t8/i4ufVG/SpR6DhAQ+iZ1btglx0jkNZiofTnUFhoeP38iToLoFXAure
xJmQIMixtHvZz9MBcazpxddQ6MPPGrN/WpQyLVGENPL2TIhLjOwajyNtm
=wY50
-----END PGP PUBLIC KEY BLOCK-----
llsirota@llsirota:~/work/study$
llsirota@llsirota:~/work/study$ gpg --list-secret-keys --keyid-format LONG
/home/llsirota/.gnupg/pubring.kbx
-----
sec   rsa4096/9977424606D4B5FB 2023-02-14 [SC]
      3594951EF0449C62EB798C709977424606D4B5FB
uid   [ultimate] llsirota0 <1132221809@pfur.ru>
ssb   rsa4096/33B916786AE73D90 2023-02-14 [E]

llsirota@llsirota:~/work/study$ git config --global commit.gpgSign true
llsirota@llsirota:~/work/study$ git config --global user.signingKey 9977424606D4
B5FB
llsirota@llsirota:~/work/study$ git config --global gpg.program $(which gpg2)
llsirota@llsirota:~/work/study$
```

Рис. 2.7: Параметры репозитория

## Настройка gh



```
llsirota@llsirota: ~/work/study
[ultimate] llsirotat0 <1132221809@pfur.ru>
ssb rsa4096/33B916786AE73D90 2023-02-14 [E]

llsirota@llsirota:~/work/study$ git config --global commit.gpgSign true
llsirota@llsirota:~/work/study$ git config --global user.signingKey 9977424606D4B5FB
llsirota@llsirota:~/work/study$ git config --global gpg.program $(which gpg2)
llsirota@llsirota:~/work/study$ 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/llsirota/.ssh/id_rsa.pub
? Title for your SSH key: GitHub CLI
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: DFEA-D2B9
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/llsirota/.ssh/id_rsa.pub
✓ Logged in as llsirotat0
llsirota@llsirota:~/work/study$
```

Рис. 2.8: Связь репозитория с аккаунтом

## Загрузка шаблона репозитория и синхронизация

```
llsirota@llsirota: ~/work/study/2022-2023/Операционные си...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Cloning into '/home/llsirota/work/study/2022-2023/Операционные системы/os-intro/
template/report'...
remote: Enumerating objects: 101, done.
remote: Counting objects: 100% (101/101), done.
remote: Compressing objects: 100% (70/70), done.
remote: Total 101 (delta 40), reused 88 (delta 27), pack-reused 0
Receiving objects: 100% (101/101), 327.25 KiB | 1.01 MiB/s, done.
Resolving deltas: 100% (40/40), done.
Submodule path 'template/presentation': checked out 'b1be3800ee91f5809264cb755d3
16174540b753e'
Submodule path 'template/report': checked out '1d1b61dcac9c287a83917b82e3aef11a3
3b1e3b2'
llsirota@llsirota:~/work/study/2022-2023/Операционные системы$
llsirota@llsirota:~/work/study/2022-2023/Операционные системы$ ls
os-intro
llsirota@llsirota:~/work/study/2022-2023/Операционные системы$ cd os-intro/
llsirota@llsirota:~/work/study/2022-2023/Операционные системы/os-intro$ ls
CHANGELOG.md  COURSE  Makefile  README.en.md  README.md
config        LICENSE  package.json  README.git-flow.md  template
llsirota@llsirota:~/work/study/2022-2023/Операционные системы/os-intro$
```

Рис. 2.9: Загрузка шаблона

## Подготовка репозитория и коммит изменений

```
llsirota@llsirota: ~/work/study/2022-2023/Операционные си...
Resolving deltas: 100% (40/40), done.
Submodule path 'template/presentation': checked out 'b1be3800ee91f5809264cb755d3
16174540b753e'
Submodule path 'template/report': checked out '1d1b61dcac9c287a83917b82e3aef11a3
3b1e3b2'
llsirota@llsirota:~/work/study/2022-2023/Операционные системы$
llsirota@llsirota:~/work/study/2022-2023/Операционные системы$ ls
os-intro
llsirota@llsirota:~/work/study/2022-2023/Операционные системы$ cd os-intro/
llsirota@llsirota:~/work/study/2022-2023/Операционные системы/os-intro$ ls
CHANGELOG.md  COURSE  Makefile  README.en.md  README.md
config        LICENSE  package.json  README.git-flow.md  template
llsirota@llsirota:~/work/study/2022-2023/Операционные системы/os-intro$
llsirota@llsirota:~/work/study/2022-2023/Операционные системы/os-intro$ rm packa
ge.json
llsirota@llsirota:~/work/study/2022-2023/Операционные системы/os-intro$ make COU
RSE=os-intro
llsirota@llsirota:~/work/study/2022-2023/Операционные системы/os-intro$ make
make: Nothing to be done for 'all'.
llsirota@llsirota:~/work/study/2022-2023/Операционные системы/os-intro$ ls
CHANGELOG.md  labs      prepare  README.en.md  template
config        LICENSE  presentation  README.git-flow.md
COURSE        Makefile  project-personal  README.md
llsirota@llsirota:~/work/study/2022-2023/Операционные системы/os-intro$
```

Рис. 2.10: Первый коммит

## **3 Вывод**

Мы приобрели практические навыки работы с сервисом github.

## 4 Контрольные вопросы

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

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

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

- хранилище - пространство на накопителе где расположен репозиторий
- commit - сохранение состояния хранилища
- история - список изменений хранилища (коммитов)
- рабочая копия - локальная копия сетевого репозитория, в которой работает программист. Текущее состояние файлов проекта, основанное на версии, загруженной из хранилища (обычно на последней)

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

Централизованные системы контроля версий представляют собой приложения типа клиент-сервер, когда репозиторий проекта существует в единственном экземпляре и хранится на сервере. Доступ к нему осуществлялся через специальное клиентское приложение. В качестве примеров таких программных продуктов можно привести CVS, Subversion.

Распределенные системы контроля версий (Distributed Version Control System, DVCS) позволяют хранить репозиторий (его копию) у каждого разработчика, работающего с данной системой. При этом можно выделить центральный репозиторий (условно), в который будут отправляться изменения из локальных и, с ним же эти локальные репозитории будут синхронизироваться. При работе с такой системой, пользователи периодически синхронизируют свои локальные репозитории с центральным и работают непосредственно со своей локальной копией. После внесения достаточного количества изменений в локальную копию они (изменения) отправляются на сервер. При этом сервер, чаще всего, выбирается условно, т.к. в большинстве DVCS нет такого понятия как “выделенный сервер с центральным репозиторием”.

#### 4. Опишите действия с VCS при единоличной работе с хранилищем.

Один пользователь работает над проектом и по мере необходимости делает коммиты, сохраняя определенные этапы.

#### 5. Опишите порядок работы с общим хранилищем VCS.

Несколько пользователей работают каждый над своей частью проекта. При этом каждый должен работать в своей ветки. При завершении работы ветка пользователя сливается с основной веткой проекта.

#### 6. Каковы основные задачи, решаемые инструментальным средством git?

- Ведение истории версий проекта: журнал (log), метки (tags), ветвления (branches).

- Работа с изменениями: выявление (diff), слияние (patch, merge).
- Обеспечение совместной работы: получение версии с сервера, загрузка обновлений на сервер.

7. Назовите и дайте краткую характеристику командам git.

- git config - установка параметров
- git status - полный список изменений файлов, ожидающих коммита
- git add . - сделать все измененные файлы готовыми для коммита.
- git commit -m "[descriptive message]" - записать изменения с заданным сообщением.
- git branch - список всех локальных веток в текущей директории.
- git checkout [branch-name] - переключиться на указанную ветку и обновить рабочую директорию.
- git merge [branch] — соединить изменения в текущей ветке с изменениями из заданной.
- git push - запустить текущую ветку в удаленную ветку.
- git pull - загрузить историю и изменения удаленной ветки и произвести слияние с текущей веткой.

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

- git remote add [имя] [url] — добавляет удалённый репозиторий с заданным именем;
- git remote remove [имя] — удаляет удалённый репозиторий с заданным именем;
- git remote rename [старое имя] [новое имя] — переименовывает удалённый репозиторий;
- git remote set-url [имя] [url] — присваивает репозиторию с именем новый адрес;

- `git remote show [имя]` — показывает информацию о репозитории.

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

Ветвление — это возможность работать над разными версиями проекта: вместо одного списка с упорядоченными коммитами история будет расходиться в определённых точках. Каждая ветвь содержит легковесный указатель HEAD на последний коммит, что позволяет без лишних затрат создать много веток. Ветка по умолчанию называется `master`, но лучше назвать её в соответствии с разрабатываемой в ней функциональностью.

#### 10. Как и зачем можно игнорировать некоторые файлы при `commit`?

Зачастую нам не нужно, чтобы Git отслеживал все файлы в репозитории, потому что в их число могут входить:

# Список литературы

1. Лекция Системы контроля версий
2. GitHub для начинающих