

# **Отчёт по лабораторной работе №2**

**Управление версиями**

Мартынов Николай Алексеевич НБИбд-02-21

# Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Вывод	9
4	Контрольные вопросы	10
	Список литературы	14

# List of Figures

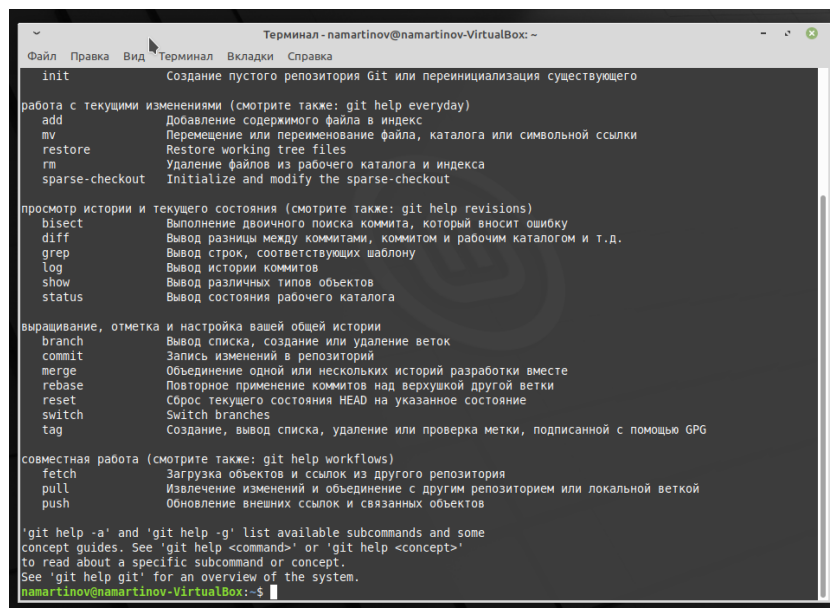
2.1	Загрузка пакетов . . . . .	5
2.2	Параметры репозитория . . . . .	5
2.3	rsa-4096 . . . . .	6
2.4	ed25519 . . . . .	6
2.5	GPG ключ . . . . .	6
2.6	GPG ключ . . . . .	7
2.7	Параметры репозитория . . . . .	7
2.8	Связь репозитория с аккаунтом . . . . .	7
2.9	Загрузка шаблона . . . . .	8
2.10	Первый коммит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

# 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.



```
Терминал - namartinov@namartinov-VirtualBox: ~
Файл  Правка  Вид  Терминал  Вкладки  Справка

Init                Создание пустого репозитория Git или переинициализация существующего

работа с текущими изменениями (смотрите также: git help everyday)
add                 Добавление содержимого файла в индекс
mv                  Перемещение или переименование файла, каталога или символической ссылки
restore             Restore working tree files
rm                  Удаление файлов из рабочего каталога и индекса
sparse-checkout     Initialize and modify the sparse-checkout

просмотр истории и текущего состояния (смотрите также: git help revisions)
bisect              Выполнение двоичного поиска коммита, который вносит ошибку
diff                Вывод разницы между коммитами, коммитом и рабочим каталогом и т.д.
grep                Вывод строк, соответствующих шаблону
log                 Вывод истории коммитов
show                Вывод различных типов объектов
status              Вывод состояния рабочего каталога

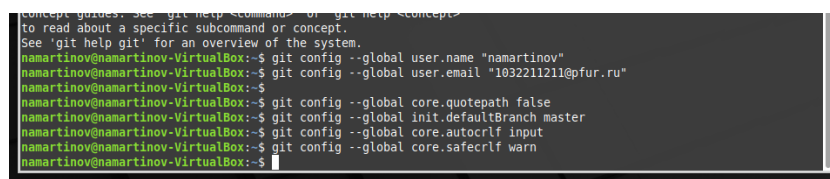
выращивание, отметка и настройка вашей общей истории
branch              Вывод списка, создание или удаление веток
commit              Запись изменений в репозиторий
merge               Объединение одной или нескольких историй разработки вместе
rebase              Повторное применение коммитов над верхушкой другой ветки
reset               Сброс текущего состояния HEAD на указанное состояние
switch              Switch branches
tag                 Создание, вывод списка, удаление или проверка метки, подписанной с помощью GPG

совместная работа (смотрите также: git help workflows)
fetch               Загрузка объектов и ссылок из другого репозитория
pull                Извлечение изменений и объединение с другим репозиторием или локальной веткой
push                Обновление внешних ссылок и связанных объектов

'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.
namartinov@namartinov-VirtualBox:~$
```

Figure 2.1: Загрузка пакетов

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



```
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.
namartinov@namartinov-VirtualBox:~$ git config --global user.name "namartinov"
namartinov@namartinov-VirtualBox:~$ git config --global user.email "1032211211@pfur.ru"
namartinov@namartinov-VirtualBox:~$ git config --global core.quotepath false
namartinov@namartinov-VirtualBox:~$ git config --global init.defaultBranch master
namartinov@namartinov-VirtualBox:~$ git config --global core.autocrlf input
namartinov@namartinov-VirtualBox:~$ git config --global core.safecrlf warn
namartinov@namartinov-VirtualBox:~$
```

Figure 2.2: Параметры репозитория

Создаем SSH ключи

```
namartinov@namartinov-VirtualBox:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/namartinov/.ssh/id_rsa):
Created directory '/home/namartinov/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/namartinov/.ssh/id_rsa
Your public key has been saved in /home/namartinov/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:3SeUTGNr075VL0s19YTF54tT3i0oHL4wL/G6is+U1CU namartinov@namartinov-VirtualBox
The key's randomart image is:
+---[RSA 4096]-----+
|
|..
|=.
|E..X*|
|.O..B%|
|.S.O@=|
|..+..O*|
|O*O..+|
|+..+..O.O|
|..+O+==|
+---[SHA256]-----+
namartinov@namartinov-VirtualBox:~$
```

Figure 2.3: rsa-4096

```
namartinov@namartinov-VirtualBox:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/namartinov/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/namartinov/.ssh/id_ed25519
Your public key has been saved in /home/namartinov/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:YHwQoqZBqWgaUDwPQBFjIethB05EnZSfdrCT+aSKo namartinov@namartinov-VirtualBox
The key's randomart image is:
+--[ED25519 256]--+
|X@*+.
|0oB+o..
|+* o ....
|+o. .o.o
|+++..o*S
|B=0...+
|+o . o
|
|E
+---[SHA256]-----+
namartinov@namartinov-VirtualBox:~$
namartinov@namartinov-VirtualBox:~$
```

Figure 2.4: ed25519

Создаем GPG ключ

```
Терминал - namartinov@namartinov-VirtualBox: ~
Файл  Правка  Вид  Терминал  Вкладки  Справка

<пр> = срок действия ключа - n месяцев
<пу> = срок действия ключа - n лет
Срок действия ключа? (0)
Срок действия ключа не ограничен
Все верно? (y/N) y

GnuPG должен составить идентификатор пользователя для идентификации ключа.
Ваше полное имя: namartinov
Адрес электронной почты: 1032211211@pfur.ru
Примечание:
Вы выбрали следующий идентификатор пользователя:
"namartinov <1032211211@pfur.ru>"

Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(O)Выход? O
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печатать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печатать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
gpg: /home/namartinov/.gnupg/trustdb.gpg: создана таблица доверия
gpg: ключ D9F4A2DC90223AD3 помечен как абсолютно доверенный
gpg: создан каталог '/home/namartinov/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/home/namartinov/.gnupg/openpgp-revocs.d/183C76317BAB37AB1DC9C278D9F4A2DC90223AD3
.rev'
открытый и секретный ключи созданы и подписаны.

pub   rsa4096 2022-04-23 [SC]
      183C76317BAB37AB1DC9C278D9F4A2DC90223AD3
uid    namartinov <1032211211@pfur.ru>
sub    rsa4096 2022-04-23 [E]

namartinov@namartinov-VirtualBox:~$
```

Figure 2.5: GPG ключ

## Добавляем GPG ключ в аккаунт

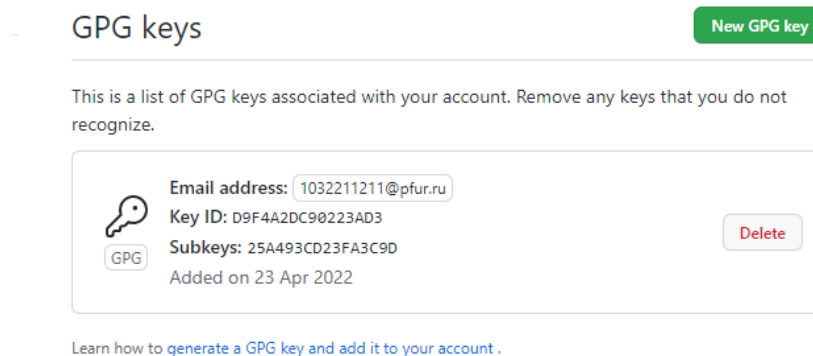


Figure 2.6: GPG ключ

## Настройка автоматических подписей коммитов git

```
bEsIRqYDsPUJqq885jmdVcQWmH1YdLce0Kcb/Do2Pmn88D6KaGoAeGZDg==
=Tzd9
-----END PGP PUBLIC KEY BLOCK-----
namartinov@namartinov-VirtualBox:~$ git config --global user.signingkey D9F4A2DC90223AD3
namartinov@namartinov-VirtualBox:~$ git config --global commit.gpgsign true
namartinov@namartinov-VirtualBox:~$ git config --global gpg.program $(which gpg2)
namartinov@namartinov-VirtualBox:~$
```

Figure 2.7: Параметры репозитория

## Настройка gh

```
namartinov@namartinov-VirtualBox:~$ git config --global commit.gpgsign true
namartinov@namartinov-VirtualBox:~$ git config --global gpg.program $(which gpg2)
namartinov@namartinov-VirtualBox:~$ 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/namartinov/.ssh/id_rsa.pub
How would you like to authenticate GitHub CLI? Login with a web browser

First copy your one-time code: 8B37-DE0C
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/namartinov/.ssh/id_rsa.pub
Logged in as namartinov
namartinov@namartinov-VirtualBox:~$
#!!! [Parent][PImageBridgeParent] Error: RunMessage(msgname=PImageBridge::Msg_WillClose) Channel closing: too late to send/recvd, messages will be lost

namartinov@namartinov-VirtualBox:~$
```

Figure 2.8: Связь репозитория с аккаунтом

## Загрузка шаблона репозитория и синхронизация

```
Терминал - namartinov@namartinov-VirtualBox: ~/work/study/2021-2022/Операционные системы
Файл  Правка  Вид  Терминал  Вкладки  Справка

namartinov@namartinov-VirtualBox:~$ mkdir -p ~/work/study/2021-2022/Операционные системы
namartinov@namartinov-VirtualBox:~$ cd ~/work/study/2021-2022/Операционные системы
namartinov@namartinov-VirtualBox:~/work/study/2021-2022/Операционные системы$ gh repo create study_2021-2022_os-intro --template=yamadharma/course-directory-student-template --public
Created repository namartinov/study_2021-2022_os-intro on GitHub
namartinov@namartinov-VirtualBox:~/work/study/2021-2022/Операционные системы$ git clone --recursive git@github.com:
namartinov/study_2021-2022_os-intro.git os-intro
Клонирование в «os-intro»...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
ECDSA key fingerprint is SHA256:p2QAMXN1C1TJYWeI0ttrVc98/R1BUFWu3/LiyKgUfQM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com,140.82.121.4' (ECDSA) to the list of known hosts.
remote: Enumerating objects: 20, done.
remote: Counting objects: 100% (20/20), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 20 (delta 2), reused 15 (delta 2), pack-reused 0
Получение объектов: 100% (20/20), 12.49 Киб | 2.08 Миб/с, готово.
Определение изменений: 100% (2/2), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистрирован по пути «template/report»
Клонирование в «/home/namartinov/work/study/2021-2022/Операционные системы/os-intro/template/presentation»...
remote: Enumerating objects: 42, done.
remote: Counting objects: 100% (42/42), done.
remote: Compressing objects: 100% (34/34), done.
remote: Total 42 (delta 9), reused 40 (delta 7), pack-reused 0
Клонирование в «/home/namartinov/work/study/2021-2022/Операционные системы/os-intro/template/report»...
remote: Enumerating objects: 78, done.
remote: Counting objects: 100% (78/78), done.
remote: Compressing objects: 100% (52/52), done.
remote: Total 78 (delta 31), reused 69 (delta 22), pack-reused 0
Подмодуль по пути «template/presentation»: забрано состояние «3eae6b7586f8a9aded2b506cd1018e625b228b93»
Подмодуль по пути «template/report»: забрано состояние «df7b2ef80f8def3b9a496f8695277469a1a7842a»
namartinov@namartinov-VirtualBox:~/work/study/2021-2022/Операционные системы$
```

Figure 2.9: Загрузка шаблона

## Подготовка репозитория и коммит изменений

```
Терминал - namartinov@namartinov-VirtualBox: ~/work/study/2021-2022/Операционные системы/os-intro
Файл  Правка  Вид  Терминал  Вкладки  Справка

namartinov@namartinov-VirtualBox:~/work/study/2021-2022/Операционные системы/os-intro$ cd ~/work/study/2021-2022/Операционные системы/os-intro
namartinov@namartinov-VirtualBox:~/work/study/2021-2022/Операционные системы/os-intro$ rm package.json
namartinov@namartinov-VirtualBox:~/work/study/2021-2022/Операционные системы/os-intro$ make COURSE=os-intro
namartinov@namartinov-VirtualBox:~/work/study/2021-2022/Операционные системы/os-intro$ git add .
namartinov@namartinov-VirtualBox:~/work/study/2021-2022/Операционные системы/os-intro$ git commit -am 'feat(main): make course structure'
[master 1c61c9f] feat(main): make course structure
16 files changed, 1580 insertions(+), 14 deletions(-)
create mode 100644 labs/lab01..15/presentation/Makefile
create mode 100644 labs/lab01..15/presentation/presentation.md
create mode 100644 labs/lab01..15/report/Makefile
create mode 100644 labs/lab01..15/report/bib/cite.bib
create mode 100644 labs/lab01..15/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab01..15/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 labs/lab01..15/report/report.md
delete mode 100644 package.json
create mode 100644 project-personal/stage1..6/presentation/Makefile
create mode 100644 project-personal/stage1..6/presentation/presentation.md
create mode 100644 project-personal/stage1..6/report/Makefile
create mode 100644 project-personal/stage1..6/report/bib/cite.bib
create mode 100644 project-personal/stage1..6/report/image/placeimg_800_600_tech.jpg
create mode 100644 project-personal/stage1..6/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 project-personal/stage1..6/report/report.md
namartinov@namartinov-VirtualBox:~/work/study/2021-2022/Операционные системы/os-intro$ git push
Перечисление объектов: 20, готово.
Подсчет объектов: 100% (20/20), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (14/14), готово.
Запись объектов: 100% (19/19), 266.46 Киб | 2.47 Миб/с, готово.
Всего 19 (изменения 2), повторно использовано 0 (изменения 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To github.com:namartinov/study_2021-2022_os-intro.git
66da20c..1c61c9f master -> master
namartinov@namartinov-VirtualBox:~/work/study/2021-2022/Операционные системы/os-intro$
```

Figure 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