

Лабораторная работа №2

Дисциплина - Операционные Системы

Азарцова В. В.

6 марта 2025

Российский университет дружбы народов, Москва, Россия

Преподаватель Кулябов Д. С.

Информация

- Азарцова Вероника Валерьевна
- НКАбд-01-24, студ. билет №1132246751
- Российский университет дружбы народов
- 1132246751@pfur.ru
- <https://github.com/vvazarcova>

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

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

Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом.

Они также могут обеспечивать дополнительные возможности: поддерживать работу с несколькими версиями одного файла, сохраняя историю изменений. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил.

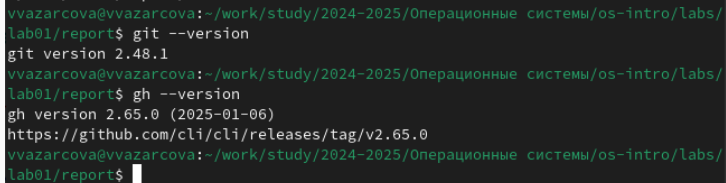
В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным.

Основные команды git:

1. `git init` (создать основное древо)
2. `git pull` (получить обновления)
3. `git push` (отправить изменения)
4. `git status` (посмотреть список измененных файлов)
5. `git diff` (посмотреть текущие изменения)
6. `git add {file_name}` (добавить измененные файлы, чтобы добавить все - .)
7. `git rm {file_name}` (удалить измененные файлы)
8. `git commit -am 'Описание коммита'` (сохранить добавленные изменения в коммит)

Выполнение лабораторной работы

У меня уже установлен git и gh, так что проверяю, что они установлены и что они последней версии.

A terminal window with a dark background and green text. The prompt is 'vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro/labs/lab01/report\$'. The first command is 'git --version', which outputs 'git version 2.48.1'. The second command is 'gh --version', which outputs 'gh version 2.65.0 (2025-01-06)' followed by the URL 'https://github.com/cli/cli/releases/tag/v2.65.0'. The prompt is then shown again.

```
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro/labs/
lab01/report$ git --version
git version 2.48.1
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro/labs/
lab01/report$ gh --version
gh version 2.65.0 (2025-01-06)
https://github.com/cli/cli/releases/tag/v2.65.0
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro/labs/
lab01/report$
```

Рис. 1: Установленный git и gh

Задаю имя и электронную почту владельца репозитория, настраиваю верификацию и подписание коммитов git и задаю имя начальной ветки - master.

```
foot
[vvazarcova@vvazarcova ~]$ git config --global user.name "Veronika Azarcova"
[vvazarcova@vvazarcova ~]$ git config --global user.email "veronika.azartsova@gmail.com"
[vvazarcova@vvazarcova ~]$ git config --global core.quotepath false
[vvazarcova@vvazarcova ~]$ git config --global init.defaultBranch master
[vvazarcova@vvazarcova ~]$ git ls-files --eol
```

Рис. 2: Имя и эл. почта владельца

Генерирую ключ SSH.

По алгоритму rsa с ключём размером 4096 бит:

```
[vvazarcova@vvazarcova ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/vvazarcova/.ssh/id_rsa): sshkey
Enter passphrase for "sshkey" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in sshkey
Your public key has been saved in sshkey.pub
The key fingerprint is:
SHA256:kmrdXDKSCaZgPMC4evSo6bEyFjHkjsbIU5NAwKrzl4 vvazarcova@vvazarcova
The key's randomart image is:
+---[RSA 4096]-----+
|=.                  |
|*... o             |
|o+ o+ o            |
|o =o.+ o           |
|o+=.o + S +        |
|@+ = o = + .       |
|+0*Eo . +          |
|o *                 |
```

По алгоритму ed25519:

```
bash: /home/vvazarcova/.ssh/id_ed25519.pub: Нет такого файла или каталога
[vvazarcova@vvazarcova ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/vvazarcova/.ssh/id_ed25519): sshkey2
Enter passphrase for "sshkey2" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in sshkey2
Your public key has been saved in sshkey2.pub
The key fingerprint is:
SHA256:cnxpi2ASDgDn7kLDUafSQ0ebN9I7Bo1c8y3nFBE90sE vvazarcova@vvazarcova
The key's randomart image is:
+--[ED25519 256]--+
| =0.O .   O=..   |
|  +* O     ..E   |
| 00+.O     .. .  |
| ..+oB.O.. ..   |
| +.Bo+=oS++     |
| ... +00+=+ .   |
| . .  +   ...   |
| . . .           |
|                 |
+-----[SHA256]-----+
```

Ключ PGP

Генерирую PGP ключ типа RSA and RSA размера 4096 байт с бесконечным сроком действия, введя свою личную информацию, и вывожу на экран список ключей.

[illegible]

Настройка подписи Git

Настраиваю автоматические подписи git используя электронную почту, которую я использовала для создания ключа, и авторизуюсь в и настраиваю gh.

```
[vvaazacova@vvaazacova ~]$ git config --global user.signingkey A080BA77E9094AD
[vvaazacova@vvaazacova ~]$ git config --global commit.gpgsign true
[vvaazacova@vvaazacova ~]$ git config --global gpg.program $(which gpg2)
[vvaazacova@vvaazacova ~]$ gh auth login
bash: gh: команда не найдена
[vvaazacova@vvaazacova ~]$ gh
bash: gh: команда не найдена
[vvaazacova@vvaazacova ~]$ sudo dnf gh
[sudo] пароль для vvaazacova:
Неизвестный аргумент "gh" для команды "dnf5". Добавьте "--help" для получения дополнительной информации об аргументах.
Это может быть команда, предоставляемая плагином, попробуйте: dnf5 install 'dnf5-command(gh)'
[vvaazacova@vvaazacova ~]$ gh
bash: gh: команда не найдена
[vvaazacova@vvaazacova ~]$ gh auth login
bash: gh: команда не найдена
[vvaazacova@vvaazacova ~]$ sudo dnf gh
Неизвестный аргумент "gh" для команды "dnf5". Добавьте "--help" для получения дополнительной информации об аргументах.
Это может быть команда, предоставляемая плагином, попробуйте: dnf5 install 'dnf5-command(gh)'
[vvaazacova@vvaazacova ~]$ sudo dnf install gh
Обновление и загрузка репозитория:
Репозитории загружены.
Пакет
```

Репозиторий	Арх.	Версия	Разм
gh	x86_64	2.65.0-1.fc41	42.6 M

```
ИБ
Сводка транзакции:
Установка: 1 пакета

Общий размер входящих пакетов составляет 10 MiB. Необходимо загрузить 10 MiB.
После этой операции будут использоваться дополнительные 43 MiB (установка 43 MiB, удаление 0 B).
Is this ok [y/N]: y

[1/1] gh-0:2.65.0-1.fc41.x86_64
100% | 1.7 MiB/s | 10.3 MiB | 0m0s
Es
-----
[1/1] Total
100% | 1.5 MiB/s | 10.3 MiB | 0m0s
Fs
Выполнение транзакции
[1/3] Проверить файлы пакета
100% | 28.0 B/s | 1.0 B | 0m0s
[2/3] Подготовить транзакцию
100% | 2.0 B/s | 1.0 B | 0m0s
[3/3] Установка gh-0:2.65.0-1.fc41.x86_64
100% | 25.0 MiB/s | 42.7 MiB | 0m0s
```

Я уже зарегистрирована на github с той электронной почты, которую использовала для двух прошлых шагов. Далее буду использовать свой уже готовый аккаунт.

Создание локального каталога для работы по предмету

Создаю шаблон рабочего пространства для предмета Операционные системы, перехожу в каталог курса и удаляю лишний файл package.json.

```
vvazarcova@vvazarcova:~$ mkdir -p ~/work/study/2022-2023/"Операционные системы"
vvazarcova@vvazarcova:~$ cd ~/work/study/2022-2023/"Операционные системы"
vvazarcova@vvazarcova:~/work/study/2022-2023/Операционные системы$ mkdir -p ~/work/study/20
24-2025/"Операционные системы"
vvazarcova@vvazarcova:~/work/study/2022-2023/Операционные системы$ cd ~/work/study/2024-202
5/"Операционные системы"
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы$ git clone --recursive gi
t@github.com:vvazarcova/study_2024-2025_os-intro.git os-intro
Клонирование в «os-intro»...
remote: Enumerating objects: 36, done.
remote: Counting objects: 100% (36/36), done.
remote: Compressing objects: 100% (35/35), done.
remote: Total 36 (delta 1), reused 21 (delta 0), pack-reused 0 (from 0)
Получение объектов: 100% (36/36), 19.38 КиБ | 413.00 КиБ/с, готово.
Определение изменений: 100% (1/1), 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/vvazarcova/work/study/2024-2025/Операционные системы/os-intro/templat
e/presentation»...
remote: Enumerating objects: 111, done.
remote: Counting objects: 100% (111/111), done.
remote: Compressing objects: 100% (77/77), done.
remote: Total 111 (delta 42), reused 100 (delta 31), pack-reused 0 (from 0)
Получение объектов: 100% (111/111), 102.17 КиБ | 223.00 КиБ/с, готово.
Определение изменений: 100% (42/42), готово.
Клонирование в «/home/vvazarcova/work/study/2024-2025/Операционные системы/os-intro/templat
e/report»...
remote: Enumerating objects: 142, done.
remote: Counting objects: 100% (142/142), done.
remote: Compressing objects: 100% (97/97), done.
remote: Total 142 (delta 60), reused 121 (delta 39), pack-reused 0 (from 0)
```


Создание локального каталога для работы по предмету

Создаю нужные каталоги и добавляю их в коммит.

```
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы$ cd os-intro
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ rm package.json
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ echo os-intro > COURSE
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ make prepare
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ git add .
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ git commit -am 'feat(main):
make course structure'
[master b36e7bf] feat(main): make course structure
405 files changed, 98413 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 labs/lab01/presentation/.projectile
create mode 100644 labs/lab01/presentation/.texlabroot
create mode 100644 labs/lab01/presentation/Makefile
create mode 100644 labs/lab01/presentation/image/kulyabov.jpg
create mode 100644 labs/lab01/presentation/presentation.md
create mode 100644 labs/lab01/report/Makefile
create mode 100644 labs/lab01/report/bib/cite.bib
```

Рис. 8: Создание рабочего пространства

Создание локального каталога для работы по предмету

Отправляю файлы на сервер.

```
v vazarcova@v vazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ git push
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), 341.67 КиБ | 1.79 МиБ/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:v vazarcova/study_2024-2025_os-intro.git
   69ce19d..b36e7bf  master -> master
v vazarcova@v vazarcova:~/work/study/2024-2025/Операционные системы/os-intro$
```

Рис. 9: Создание рабочего пространства

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

VCS (Version Control System) — это система управления изменениями в файлах и коде. Она позволяет отслеживать историю изменений, управлять версиями, работать в команде и откатываться к предыдущим состояниям.

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

- хранилище (репозиторий) — место, где хранятся все версии проекта
- commit — фиксированное изменение в коде с комментарием
- история — последовательность commit'ов, отражающая эволюцию проекта
- рабочая копия — текущая версия файлов, с которой работает разработчик

3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.
- централизованные VCS (CVCS): один сервер содержит репозиторий, разработчики получают и отправляют изменения туда. Пример: SVN
 - децентрализованные VCS (DVCS): каждый разработчик имеет полный репозиторий, можно работать без подключения к серверу. Пример: Git, Mercurial
4. Опишите действия с VCS при единоличной работе с хранилищем.
- Создать репозиторий
 - Добавить файлы и сделать commit
 - Вносить изменения и фиксировать их (commit'ы)
 - Использовать историю для отката при необходимости
 - Создавать ветки для изоляции работы

5. Опишите порядок работы с общим хранилищем VCS.
- Клонировать (скопировать) удалённый репозиторий
 - Внести изменения, сделать commit
 - Обновить локальные данные (pull)
 - Разрешить возможные конфликты
 - Отправить изменения в репозиторий (push)
 - Использовать ветки для разделения работы

6. Каковы основные задачи, решаемые инструментальным средством Git?

- Хранение версий проекта
- Работа с ветками и параллельной разработкой
- Совместная работа команды
- Откат и восстановление изменений
- Работа с локальными и удалёнными репозиториями

7. Назовите и дайте краткую характеристику командам Git.

- `git init` — создать новый репозиторий
- `git clone` — клонировать репозиторий с сервера
- `git add` — добавить файлы в область подготовки
- `git commit` — сохранить изменения
- `git status` — проверить статус файлов
- `git log` — показать историю коммитов
- `git branch` — работа с ветками
- `git checkout` или `git switch` — переключение между ветками
- `git merge` — объединение веток
- `git pull` — получение изменений из репозитория на сервере
- `git push` — отправка изменений в репозиторий на сервере

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

- Локальный репозиторий:

```
git init
```

```
git add .
```

```
git commit -m "Первый коммит"
```

- Удалённый репозиторий:

```
git clone https://github.com/user/repo.git
```

```
git pull origin main
```

```
git push origin main
```


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

Ветви — это разные линии разработки в репозитории. С ними можно:

- Работать над новыми функциями не меняя основного кода
- Параллельно создавать новые возможности
- Объединять изменения с основной веткой только после тестирования

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

Для исключения временных, служебных и ненужных файлов. Чтобы это сделать, создаётся файл **.gitignore**, где указываются файлы и каталоги, которые не должны быть в репозитории.

Выводы

Что мне удалось:

- Получить практические навыки в работе с системой контроля версий git
- Создать и настроить рабочее пространство для выполнения заданий по дисциплине
Операционные системы.

Если вам понравилось - посмотрите остальные мои презентации!