

Отчёт по лабораторной работе №2

Управление версиями

Амира Хакимова

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Вывод	17
4	Контрольные вопросы	18

Список иллюстраций

2.1	Загрузка пакетов	7
2.2	Параметры репозитория	8
2.3	rsa-4096	9
2.4	ed25519	10
2.5	GPG ключ	11
2.6	GPG ключ	12
2.7	Параметры репозитория	13
2.8	Связь репозитория с аккаунтом	14
2.9	Загрузка шаблона	15
2.10	Первый коммит	16

Список таблиц

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.

```
aeihakimova@aeihakimova:~$ git
использование: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
    [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
    [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--no-lazy-fetch]
    [--no-optional-locks] [--no-advice] [--bare] [--git-dir=<path>]
    [--work-tree=<path>] [--namespace=<name>] [--config-env=<name>=<envvar>]
    <command> [<args>]
```

Стандартные команды Git используемые в различных ситуациях:

создание рабочей области (смотрите также: `git help tutorial`)

<code>clone</code>	Клонирование репозитория в новый каталог
<code>init</code>	Создание пустого репозитория Git или переинициализация существующего

работа с текущими изменениями (смотрите также: `git help everyday`)

<code>add</code>	Добавление содержимого файла в индекс
<code>mv</code>	Перемещение или переименование файла, каталога или символической ссылки
<code>restore</code>	Восстановление файлов в рабочем каталоге
<code>rm</code>	Удаление файлов из рабочего каталога и индекса

просмотр истории и текущего состояния (смотрите также: `git help revisions`)

<code>bisect</code>	Выполнение двоичного поиска коммита, который вносит ошибку
<code>diff</code>	Вывод разницы между коммитами, коммитом и рабочим каталогом и т.д.
<code>grep</code>	Вывод строк, соответствующих шаблону
<code>log</code>	Вывод истории коммитов
<code>show</code>	Вывод различных типов объектов
<code>status</code>	Вывод состояния рабочего каталога

Рис. 2.1: Загрузка пакетов

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

```
aehakimova@aehakimova:~$ git config --global user.name "amirahakimova"
aehakimova@aehakimova:~$ git config --global user.email "1132243103@rudn.university"
aehakimova@aehakimova:~$ git config --global core.quotepath false
aehakimova@aehakimova:~$ git config --global init.defaultBranch master
aehakimova@aehakimova:~$ git config --global core.autocrlf input
aehakimova@aehakimova:~$ git config --global core.safecrlf warn
aehakimova@aehakimova:~$ !
```

Рис. 2.2: Параметры репозитория

Создаем SSH ключи


```

aehakimova@aehakimova:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/aehakimova/.ssh/id_rsa):
Created directory '/home/aehakimova/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/aehakimova/.ssh/id_rsa
Your public key has been saved in /home/aehakimova/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:LzFfNoWdQqLZkb8IKevRKVMiXfL7DWFw4GnoxWT52s aehakimova@aehakimova
The key's randomart image is:
+---[RSA 4096]-----+
|          +=...|
|      . +=+.+0|
|    . +=00+.0|
|    . +.B0+. .|
|    .S*.+0 ..|
|    ==+00..E|
|    ..+0. 0.|
|    .. . .|
|          |
+----[SHA256]-----+
aehakimova@aehakimova:~$ █

```

Рис. 2.3: rsa-4096

```

aehakimova@aehakimova:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/aehakimova/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/aehakimova/.ssh/id_ed25519
Your public key has been saved in /home/aehakimova/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:D219Q7mrH/RGVht0gJG0kECDlf6v8azJUraQTRBy0Ak aehakimova@aehakimova
The key's randomart image is:
+--[ED25519 256]--+
| E..*B+ . .+.0.|
| =00.0 0 ...|
| 0 . . 0 0..|
| . .... .+|
| =S 0 . =.0|
| 0 =+ 0 *|
| +.0. 0 0|
| ...=. . 0|
| .00 ...|
+----[SHA256]-----+
aehakimova@aehakimova:~$ █

```

Рис. 2.4: ed25519

Создаем GPG ключ

```

aehakimova@aehakimova:~$ gpg --full-generate-key
gpg (GnuPG) 2.4.5; Copyright (C) 2024 g10 Code GmbH
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: создан каталог '/home/aehakimova/.gnupg'
Выберите тип ключа:
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) *default*
 (10) ECC (только для подписи)
 (14) Existing key from card
Ваш выбор? 1
длина ключей RSA может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072) 4096
Запрошенный размер ключа - 4096 бит
Выберите срок действия ключа.
  0 = не ограничен
  <n> = срок действия ключа - n дней
  <n>w = срок действия ключа - n недель
  <n>m = срок действия ключа - n месяцев
  <n>y = срок действия ключа - n лет
Срок действия ключа? (0) 0
Срок действия ключа не ограничен
Все верно? (y/N) y

```

Рис. 2.5: GPG ключ

Добавляем GPG ключ в аккаунт

```

aehakimova@aehakimova:~$ gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: глубина: 0 достоверных: 1 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f, 1u
[keyboard]
-----
sec   rsa4096/3314511D52086862 2025-02-13 [SC]
      DD6D5E4AB971227C246E9483314511D52086862
uid   [ абсолютно ] amirahakimova <1132243103@rudn.university>
ssb   rsa4096/DCD42CD992A95560 2025-02-13 [E]

aehakimova@aehakimova:~$ gpg --armor --export 3314511D52086862
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGet1iyBEADZGh2yAEXj1DeRBUUklEbtFK4w6Hr08im38KpvVbKbfhEAL7+
Ym5IZYCTSoqqtuSGcvh/9gT+LTKNVof5vbhwQbrAT1X106JeYq6H9+F9A8eaxD80
NPHcRe/tIVjkzriP/u29P7nImoQLQJ7uePC4JtCQt+Z82AkJoLCWtJv1Pt1QUJ8f
3F6X4M+Z+Jb2CRUaEpXf1o+No4dI77jNXpGUJifRtEnQWCS/L91NEumAdWvJOH50
ZnA9s3XxrIh5P/e1uEiHYKYZp0PI3uq4KYGpaeByDY8J1aaZmb4qhmyNzcoHBY+o
/VVsasNuCB3BnUjkb2M6joCY8NoFi7ryiLss7Gy/X3e090nnt5EQbUvk0aU0qRE0
5mISMxbC2yZH0eJnME9EyRQ3eK5FFInwtcGjA7BdTu9s1rrtIyW7jVZXgkVJBz6N
U5uxYy7zTbmQEpl0zFxV46RPuvDHG9RbTsERwiELuif7W4zQyIN5/LFpBtAyFIhL
A+pqRDXC0uH+QYzJenLAUqCLWtr++WZW7hBXBLbsF/KtDdgsS6MBjcu/aigawvD
uwGMU70xjYaI2bs+MQFFXJaRtqeUBebvUy2r2gQsZoZZStT7DU4ihTaA3U9huJJy
rIg31/7/coW6Vm8vpDUS8saJnR5n6mwYoTPdU4KPb96veU392+026/7/rwARAQAB
tCphbWlyYWWha2ltb3ZhIDwxMTMyMjQzMTA3QzJlZG4udW5pdmVyc2l0eT6JAlEE
EwEIAQsWIQTdbV5KuXEifCRu1IMzFFEdUghoYgUCZ63WJgIbAwULCQgHAgIiAgYV
CgkICwIEFgIDAQIeBwIXgAAKCRAzFFEdUghoYltdD/9v29Y/qGzLU9JaiiXx9/SU
8iiVo1zWu70vuuu09iCYK04V0es7C16fXPskp6bEaPF09EUGR7nkr0TWE8S2vWhr

```

Рис. 2.6: GPG ключ

Настройка автоматических подписей коммитов git

```

0HbfuYfPGGQnHkfyLQ6IVhRvfgL/NRaisc7ozDXThzGpGVySpxb5Lipf9wLUS0
k1t56xKz50g+u0l+vI6BVEoBgP/ooBCF9WsxUHDpfQMX0/sc2Me9gA1VA16ynlih
iKxbB/Mt9kM1HN0M8ES0TWuaAzKiDdU257E0rIHT3ojzcUr8Nzn0BFpsYoeiesvE
HBWkJKF0ic6T4f0bVwZak370XArm3RQGC5FBu8Hr1kY6EimMRga87Br/V3Za5pW1
0LdTviFwC28LbXsJWxvdktmCMXWr09oErk0FSbvpQ7dVKUHc09j0gTSXdgV6nuBI
c0ia4XRgosHbA86KuNKqLNgNQFpwnQU0d70dukmvQRXyr0oRpbW62EF2auXTY0p0
yi1Crz+ik4jE2EU/E+uq5BsikPw12dT0Jz6QjLd5eoFcIhGZsngJA24cxLIzeW7Y
vMnNjJrGLiU0tbRGaDZ8Rckio5NBm1xzRoMXGkKZwTm76DdEsP7WAUqEivfLYJS
zI9DdG7/iALnZY0NmyzZiBft7ATL2YiUs5SAs2TzyvQfWq9Tdcvm2XjGyafDf4Ho
kR1mgv2inpaKDZ7MH87DmSZnJb0SnRny53vHw4TMbwOPPFdzfgMHTKB+0bvaf6Is
PzlsA9yX1I6Q
=lDaJ
-----END PGP PUBLIC KEY BLOCK-----
aehakimova@aehakimova:~$ git config --global user.signingkey 3314511D52086862
aehakimova@aehakimova:~$ git config --global commit.gpgsign true
aehakimova@aehakimova:~$ git config --global gpg.program $(which gpg2)
aehakimova@aehakimova:~$

```

Рис. 2.7: Параметры репозитория

Настройка gh

```
aehakimova@aehakimova:~$  
aehakimova@aehakimova:~$ gh auth login  
? Where do you use GitHub? GitHub.com  
? What is your preferred protocol for Git operations on this host? SSH  
? Upload your SSH public key to your GitHub account? /home/aehakimova/.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: B4BA-9B51  
Press Enter to open https://github.com/login/device 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/aehakimova/.ssh/id_rsa.pub  
✓ Logged in as amirahakimova  
aehakimova@aehakimova:~$
```

Рис. 2.8: Связь репозитория с аккаунтом

Загрузка шаблона репозитория и синхронизация

```

aehakimova@aehakimova:~$ mkdir -p ~/work/study/2024-2025/"Операционные системы"
aehakimova@aehakimova:~$ cd ~/work/study/2024-2025/"Операционные системы"
aehakimova@aehakimova:~/work/study/2024-2025/Операционные системы$ gh repo create os-intro --template=yamadharma/course-directory-student-template --public
✓ Created repository amirahakimova/os-intro on GitHub
https://github.com/amirahakimova/os-intro
aehakimova@aehakimova:~/work/study/2024-2025/Операционные системы$ git clone --recursive git@github.com:amirahakimova/os-intro.git os-intro
Клонирование в «os-intro»...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhbpZisF/zLDA0zPMSvHdKr4UvCOqU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
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), 19.37 Киб | 19.38 Миб/с, готово.
Определение изменений: 100% (1/1), готово.

```

Рис. 2.9: Загрузка шаблона

Подготовка репозитория и коммит изменений

```
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_fignos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_secnos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/core.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/main.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 project-personal/stage6/report/report.md
aehakimova@aehakimova:~/work/study/2024-2025/Операционные системы/os-intro$ git push
Перечисление объектов: 38, готово.
Подсчет объектов: 100% (38/38), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (37/37), 342.27 Киб | 2.21 Миб/с, готово.
Total 37 (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:amirahakimova/os-intro.git
   7fb93fa..2c6d6de master -> master
aehakimova@aehakimova:~/work/study/2024-2025/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: