

Отчёт по лабораторной работе №2

Управление версиями

Алиев Руслан Нияз оглы

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.

```
ruslanaliev@ruslanaliev:~$ 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.

```
ruslanaliyev@ruslanaliyev:~$  
ruslanaliyev@ruslanaliyev:~$ git config --global user.name "AliyevRuslan-rudn"  
ruslanaliyev@ruslanaliyev:~$ git config --global user.email "1132247533@rudn.university"  
ruslanaliyev@ruslanaliyev:~$ git config --global core.quotepath false  
ruslanaliyev@ruslanaliyev:~$ git config --global init.defaultBranch master  
ruslanaliyev@ruslanaliyev:~$ git config --global core.autocrlf input  
ruslanaliyev@ruslanaliyev:~$ git config --global core.safecrlf warn  
ruslanaliyev@ruslanaliyev:~$
```

Рис. 2.2: Параметры репозитория

Создаем SSH ключи


```

ruslanaliev@ruslanaliev:~$
ruslanaliev@ruslanaliev:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ruslanaliev/.ssh/id_rsa):
Created directory '/home/ruslanaliev/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ruslanaliev/.ssh/id_rsa
Your public key has been saved in /home/ruslanaliev/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:+HvVTDvLSXzldG/EoXVhXugItj+/2X+6tTlAJ6blw0c ruslanaliev@ruslanaliev
The key's randomart image is:
+---[RSA 4096]-----+
|           +o|
|          o  +oo|
|         ..o 0000|
|        . .o.B0.*|
|       . S .X=0=+|
|      . . +EB =|
|     . . =.*.|
|    .. ==+|
|   .. ==*|
+-----[SHA256]-----+
ruslanaliev@ruslanaliev:~$

```

Рис. 2.3: rsa-4096

```

ruslanaliev@ruslanaliev:~$
ruslanaliev@ruslanaliev:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ruslanaliev/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ruslanaliev/.ssh/id_ed25519
Your public key has been saved in /home/ruslanaliev/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:3m9/iXALq15cso0R7aK+1U47LxEwd2s/632b8lnVMXM ruslanaliev@ruslanaliev
The key's randomart image is:
+--[ED25519 256]--+
|
|      o . . |
|      + . .+E|
|      o. o *|
|      So oo.. o|
|      . .*+. o.|
|      .+.0* o =|
|      o ==++ oB|
|      .+oo.+o=B+|
+----[SHA256]-----+
ruslanaliev@ruslanaliev:~$

```

Рис. 2.4: ed25519

Создаем GPG ключ

```
ruslanaliev@ruslanaliev:~  
  
<p>w = срок действия ключа - n недель  
<p>m = срок действия ключа - n месяцев  
<p>y = срок действия ключа - n лет  
Срок действия ключа? (0) 0  
Срок действия ключа не ограничен  
Все верно? (y/N) y  
  
GnuPG должен составить идентификатор пользователя для идентификации ключа.  
  
Ваше полное имя: AlievRuslan-rudn  
Адрес электронной почты: 1032244920@rudn.university  
Примечание:  
Вы выбрали следующий идентификатор пользователя:  
    "AlievRuslan-rudn <1032244920@rudn.university>"  
  
Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)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/ruslanaliev/.gnupg/trustdb.gpg: создана таблица доверия  
gpg: создан каталог '/home/ruslanaliev/.gnupg/openpgp-revocs.d'  
gpg: сертификат отзыва записан в '/home/ruslanaliev/.gnupg/openpgp-revocs.d/F24F78E562BBB58EF2EAAF5FA9142  
6819697E700.rev'.  
открытый и секретный ключи созданы и подписаны.  
  
pub  rsa4096 2025-02-20 [SC]  
      F24F78E562BBB58EF2EAAF5FA91426819697E700  
uid          AlievRuslan-rudn <1032244920@rudn.university>  
sub  rsa4096 2025-02-20 [E]  
  
ruslanaliev@ruslanaliev:~$
```

Рис. 2.5: GPG ключ

Добавляем GPG ключ в аккаунт

```

ruslanaliev@ruslanaliev:~$
ruslanaliev@ruslanaliev:~$ 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/A91426819697E700 2025-02-20 [SC]
      F24F78E562BBB58EF2EAAF5FA91426819697E700
uid          [ абсолютно ] AlievRuslan-rudn <1032244920@rudn.university>
ssb  rsa4096/9139E062FEEA6AE9 2025-02-20 [E]

ruslanaliev@ruslanaliev:~$
ruslanaliev@ruslanaliev:~$ gpg --armor --export A91426819697E700
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGe3OacBEAC6dR3B9PDYfsb2zqdsksobgDy5oT9MmxgsAFSs/y6Vw+jp4J68
H8aqqkbGaPV5tLKqQkiALTBQjh8lEj8Sbu0T72GB/Q6LUvw5ldhoJpvo0ZdK92j
BTnUyeL20QxdNs9HiR1kXr3bPpb8gvKvXYk3X5B4EM9m46wPBUEjrs4v/oZPg/Re
U3W00CvuahhPzy4QfRJvcBywZ6gK4R6tJn82b0TRETj5s4J6I++aKGODETjFaYO
y0POLXcX++6+PWPouxbyLAq7rNkFl7+qdVi9PhXlynhYiDfRGvCWNGnhvHCRhxVm
A0gq15GGRJ8w8f/si1ZWaPlvQq9CiHfJ27tSdY/TcyiXq76daP04c3JRjKkMnngZ
Xla//tv/CZbusDmC0HvpE00qfs8g6T58yW++EkDuIKRpF4H58xohao8UiQbp+rpL
EbyHcJ1l5XcEFU1czk7Vy5nUziHhX0gaQngJadh0YsMhJcLKMNLCurkxnc/QoVXf
GGjEwg1rLbNCt0XGEJSzLREmr8sjfbdcaMWhIFwt3qzwqFcPwS1dVHfsaxf6/VW/
mjF10hVozSmhHBgzdYNwOqtvCxninCRk3TKPQ038ThRk6dbggaw7N1ouhzQbQRNJ
B62dkub7g/Ko0ESohazW70HwQoclvx3u46uF/agIAi76nRodeZ3d2u5qdwARAQAB
tC1BbGlldlJ1c2xhbW1ydWRuIDwxMDMyMjQ0TiwQJ1ZG4udW5pdmVyc2l0eT6J
A1EEEWIADsWIQTyT3jlyru1jvLqr1+pFCaBlpfnaAUCZ7c5pwIbAwULCQgHAgTi
AgYVCgkICwIEFgIDAQIeBwIXgAAKCRCpFCaBlpfnaC+QD/4yQ3x00m11Wgx3wxUe
GrLZcaPyP/k6qif0zFMdAZJQnItM6KeUcqhes8edEGD/uxtJjXoEoHyB1TjowMAZ
-----

```

Рис. 2.6: GPG ключ

Настройка автоматических подписей коммитов git

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Zw5quPvFGb41a2gcBgydqJD51Viti7KUaE4IgPkDlVwgepHQzPxt5hI0h3xjw1G0e
/gkbY0Xk840cPM6M3oAV0WqJKgnc6DZGKRvGDG4VZe98tNF4xuiQQQcHbYeJHqi3
B7zF2kJS5wbW+E2lyKGePvVeT2ueRaR8taadBJcD+Yy6yNmd8nXJF8aivWBAoIA8Z
6SP3yNWDbqr84qJl0xdJoiMfMvglkYdceJA8tjIe1QNkd7c131NOjEM2rCcbJkSD
KoxdFj20R7jE7uY88TENOMX1AOYk9mMT4ZVbPuP/7Q04ArnEcnfUYkuX5NfnTkXg
hzTlV5lv1Z3rCxt6w/eSAUExyGcmru/QjKudHDZB7HitscWGrnLYq0aFCAkyeJj
s0aptGg0l/SmTNuzzKpQq3hU0wX4Kz9MgQwv6apPFDGMoW9rGyMPgsTcgRbW/Rj
CCs5fAeB1MR6xhVkmHi1FZ0BN09nN15aAnj4h4s3sd8ZePF7FdLJVLRQNNUH5DkN
VIKPAySz3jr3ddE
=EZ3w
```

-----END PGP PUBLIC KEY BLOCK-----

```
ruslanaliev@ruslanaliev:~$
ruslanaliev@ruslanaliev:~$
ruslanaliev@ruslanaliev:~$ git config --global user.signingkey A91426819697E700
ruslanaliev@ruslanaliev:~$ git config --global commit.gpgsign true
ruslanaliev@ruslanaliev:~$ git config --global gpg.program $(which gpg2)
ruslanaliev@ruslanaliev:~$
```

Рис. 2.7: Параметры репозитория

Настройка gh

```
ruslanaliev@ruslanaliev:~$  
ruslanaliev@ruslanaliev:~$ 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/ruslanaliev/.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: 0670-71E9  
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/ruslanaliev/.ssh/id_rsa.pub  
✓ Logged in as AlievRuslan-rudn  
ruslanaliev@ruslanaliev:~$
```

Рис. 2.8: Связь репозитория с аккаунтом

Загрузка шаблона репозитория и синхронизация

```
ruslanaliev@ruslanaliev:~$  
ruslanaliev@ruslanaliev:~$ mkdir -p ~/work/study/2024-2025/"Операционные системы"  
ruslanaliev@ruslanaliev:~$ cd ~/work/study/2024-2025/"Операционные системы"  
ruslanaliev@ruslanaliev:~/work/study/2024-2025/Операционные системы$ gh repo create os-intro --template=y  
amadharma/course-directory-student-template --public  
✓ Created repository AlievRuslan-rudn/os-intro on GitHub  
https://github.com/AlievRuslan-rudn/os-intro  
ruslanaliev@ruslanaliev:~/work/study/2024-2025/Операционные системы$ git clone --recursive git@github.com  
:AlievRuslan-rudn/os-intro.git os-intro  
Клонирование в «os-intro»...  
The authenticity of host 'github.com (140.82.121.3)' can't be established.  
ED25519 key fingerprint is SHA256:+DiY3wvV6Tuj3hbpZisF/zLDA0zPMSvHdkr4UvCOqU.  
This key is not known by any other names.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? 
```

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
ruslanaliev@ruslanaliev:~/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), 342.28 КБ | 2.43 МБ/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:AlievRuslan-rudn/os-intro.git
   0d4498b..b35a4e8  master -> master
ruslanaliev@ruslanaliev:~/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: