

Отчёт по лабораторной работе №2

Управление версиями

Мухамметназар Турсунов

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.

```
mtursunov@mtursunov:~$ 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.

```
mtursunov@mtursunov:~$  
mtursunov@mtursunov:~$ git config --global user.name "mtursunov"  
mtursunov@mtursunov:~$ git config --global user.email "103224472@rudn.university"  
mtursunov@mtursunov:~$ git config --global core.quotepath false  
mtursunov@mtursunov:~$ git config --global init.defaultBranch master  
mtursunov@mtursunov:~$ git config --global core.autocrlf input  
mtursunov@mtursunov:~$ git config --global core.safecrlf warn  
mtursunov@mtursunov:~$
```

Рис. 2.2: Параметры репозитория

Создаем SSH ключи


```

mtursunov@mtursunov:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/mtursunov/.ssh/id_rsa):
Created directory '/home/mtursunov/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/mtursunov/.ssh/id_rsa
Your public key has been saved in /home/mtursunov/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:wdKyJLjqQzERQ6yZQguT90fK9wi0r87cAeWR+QAB3M4 mtursunov@mtursunov
The key's randomart image is:
+---[RSA 4096]-----+
|==+.                |
|+*.+ 0.0           |
|+=BoBo+ +         |
|=0 E=*0+ .        |
|. +0.+00S         |
| +... . .         |
|. 0..             |
|. + ...           |
| +B..            |
+---[SHA256]-----+
mtursunov@mtursunov:~$

```

Рис. 2.3: rsa-4096

```

mtursunov@mtursunov:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/mtursunov/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/mtursunov/.ssh/id_ed25519
Your public key has been saved in /home/mtursunov/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:P7L97DvuLSCscI+nlnWSwbl6QYyQQWkzvG6bI3UpfQg mtursunov@mtursunov
The key's randomart image is:
+--[ED25519 256]--+
|  oo+          |
|  0            |
|  . = + .      |
|  E . *        |
|  . o =S+      |
|  * * X.o      |
|  o B O.=o.    |
|  . + * ++ oo. |
|  . o.+ . =B+. |
+----[SHA256]-----+
mtursunov@mtursunov:~$

```

Рис. 2.4: ed25519

Создаем GPG ключ

```

mtursunov@mtursunov:~$
mtursunov@mtursunov:~$ 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/mtursunov/.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

```
mtursunov@mtursunov:~  
sub  rsa4096 2025-02-14 [E]  
  
mtursunov@mtursunov:~$ 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  
[keyboxd]  
-----  
sec  rsa4096/854AC9BC8521D5DB 2025-02-14 [SC]  
      82923D52FB16F0E9B1AB47F7854AC9BC8521D5DB  
uid  [ абсолютно ] mtursunov <1032244472@rudn.university>  
ssb  rsa4096/294050E0FE61B600 2025-02-14 [E]  
  
mtursunov@mtursunov:~$  
mtursunov@mtursunov:~$  
mtursunov@mtursunov:~$ gpg --armor --export 854AC9BC8521D5DB  
-----BEGIN PGP PUBLIC KEY BLOCK-----  
  
mQINBGeVkk4BEADICqQksfeID5L56SOMivYeAhERm4iJwRwyBABcED7mcNKjg0rL  
nLNmA+a8yX00Um1K4qQN/5cyiUCKmsYJ11sJ8RR+1GQsPkCz0U1K7d4qdG9yE0qV  
Iugo5zQXdIaXz4T+E7+OKu+jICHp5naMeE7cKTTD0eCWSi3nAGBzMoBem0HivPi6  
ivwXi7xh0DApY0p1g80XXMi fwJ0qUEmFk2brDniE4UUHQmp2VlRBRmIDoC7dX0Y  
4936VWXRJnLp7vAbR8QGzZsFPhsdUTUYf0gIPrwjyick7D7y7LdE48E1+3WnDvTP  
/u+IgsIC3Crg0DC2cj7nV9zLzKINaEFTjH2ANccsVFky0+TYU4q7joIqUsHBjAI8  
n5eNezCWCaY70LzGA25Kvgs3MGEE3tW10KJxWGoEBZ+6unPxr0kwpGVf7K3c1mYt  
0ldGwLxVoP1IW4PBjRS0PA302sTDR0RIEZPZ1banU13yeeYoWrLifwYF87yJrIf6  
w+o/0bkAfxcrvYK6DUJ72HUu01bP6jhJL60PSGrmmMzipreU/3cnTQzWX0Bt0x00  
Z5NWY3s8TU5Vkp3MMDAoTubZlaH+v675IQ1Xayya19Irp5CgV/yGYVRXq7KrbDM  
7sYGjXPjnZJZL81lgKdYs/FbipBcJYX+6DJXKrS03CEY1sSAhAv0/Qv9QARAQAB  
tCZtdHVyc3Vub3YgPDEwMzJAcnVkb151bml2ZXJzaXR5PokCUQQAQGA  
OxYhBIKSPVL7FvDpsath94VKybyFIIdXbBQJnryp0AhsDBQsJCAACAiICBhUKCQGL  
AoOwAaMRAh4HAheAAAQ7FTVKuhvFTdXhcnkP/3fPrY90SIIkTn7agAlunioKqG1?
```

Рис. 2.6: GPG ключ

Настройка автоматических подписей коммитов git

```

1XkRM26B9FW/H0xn74nmt/tuyE8ghoYpMuVe04c9haxlcGp4MTSZveX42S9d5mRm
8evAjoF8ZykLKUFoh0KNzs1+v2MDUya7iqBEgZpxEJoKudf5ZXS9KLfKI7mq7K
Nte9NkpDMoP0oFXehlaxVtV9noviI1h0aA7RhSxhfRCIxq/x2gND0jExL8QwA+hP
EDQihdSE2iRC/5Uha47kYIHAWp/ddiD464jztnPtoKwnfhF0btQaxw5tsmkP6Lef
F925IGtP6EELUYfiV2C9p7BKBuy9o1ePdY4oKbBVvc9v0lAH8txfm0GjH+vVDXmv
kW5l610P6B5sDSClYQlX4bst63UG9iZhkh7jRdFKdLqy0wvAvMHngN/Bc87DNTVq
KdWEiTJTLmSwi5XzBEavH3tzQjAQ75XjxxPbdfTZPXLoH9L/kQEgtR0yUbm/tP5
CIQNM8IVP0sP+GmssvJ++woUsIJG8M1rsfqWjk08Rll5J+jg8S2iohJ0TQd/0bBu
vIQoTXw=
=ghNi
-----END PGP PUBLIC KEY BLOCK-----

mtursunov@mtursunov:~$
mtursunov@mtursunov:~$ git config --global user.signingkey 854AC9BC8521D5DB
mtursunov@mtursunov:~$ git config --global commit.gpgsign true
mtursunov@mtursunov:~$ git config --global gpg.program $(which gpg2)
mtursunov@mtursunov:~$

```

Рис. 2.7: Параметры репозитория

Настройка gh

```
mtursunov@mtursunov:~$  
mtursunov@mtursunov:~$ 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/mtursunov/.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: 5BD5-455B  
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/mtursunov/.ssh/id_rsa.pub  
✓ Logged in as mtursunov  
mtursunov@mtursunov:~$
```

Рис. 2.8: Связь репозитория с аккаунтом

Загрузка шаблона репозитория и синхронизация

```
mtursunov@mtursunov:~$  
mtursunov@mtursunov:~$ mkdir -p ~/work/study/2024-2025/"Операционные системы"  
mtursunov@mtursunov:~$ cd ~/work/study/2024-2025/"Операционные системы"  
mtursunov@mtursunov:~/work/study/2024-2025/Операционные системы$ gh repo create os-intro --templ  
ate=yamadharma/course-directory-student-template --public  
✓ Created repository mtursunov/os-intro on GitHub  
https://github.com/mtursunov/os-intro  
mtursunov@mtursunov:~/work/study/2024-2025/Операционные системы$ git clone --recursive git@github  
b.com:mtursunov/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:+DiY3wvV6TuJJhpZisF/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 100644 project-personal/stage6/report/bib/cite.bib
create mode 100644 project-personal/stage6/report/image/placeimg_800_600_tech.jpg
create mode 100644 project-personal/stage6/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_eqnos.py
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
mtursunov@mtursunov:~/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 | 2.48 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:mtursunov/os-intro.git
   572e248..78fb3a2  master -> master
mtursunov@mtursunov:~/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: