

Отчёт по лабораторной работе №2

Управление версиями

Булат Исаев

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Вывод	10
4	Контрольные вопросы	11

Список иллюстраций

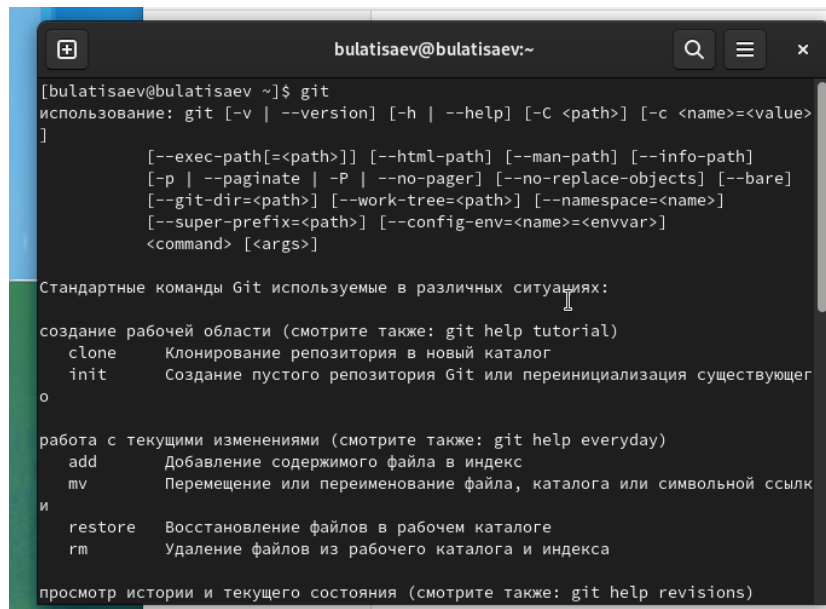
2.1	Загрузка пакетов	5
2.2	Параметры репозитория	5
2.3	rsa-4096	6
2.4	ed25519	6
2.5	GPG ключ	7
2.6	GPG ключ	7
2.7	Параметры репозитория	8
2.8	Связь репозитория с аккаунтом	8
2.9	Загрузка шаблона	9
2.10	Первый коммит	9

1 Цель работы

Целью данной работы является изучение идеологии и применения средств контроля версий и освоение умений работать с git.

2 Выполнение лабораторной работы

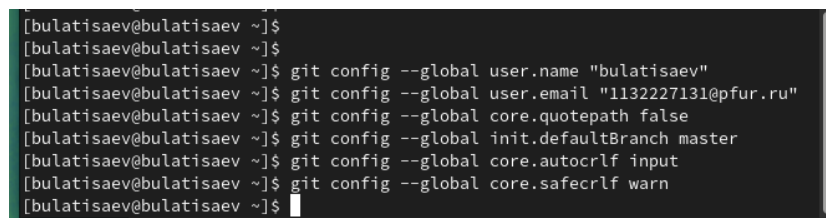
Устанавливаем git, git-flow и gh.



```
[bulatisaev@bulatisaev ~]$ 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] [--bare]  
    [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]  
    [--super-prefix=<path>] [--config-env=<name>=<envvar>]  
    <command> [<args>]  
]  
  
Стандартные команды Git используемые в различных ситуациях:  
  
создание рабочей области (смотрите также: git help tutorial)  
    clone    Клонирование репозитория в новый каталог  
    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  
  
и  
    restore  Восстановление файлов в рабочем каталоге  
    rm       Удаление файлов из рабочего каталога и индекса  
  
просмотр истории и текущего состояния (смотрите также: git help revisions)
```

Рис. 2.1: Загрузка пакетов

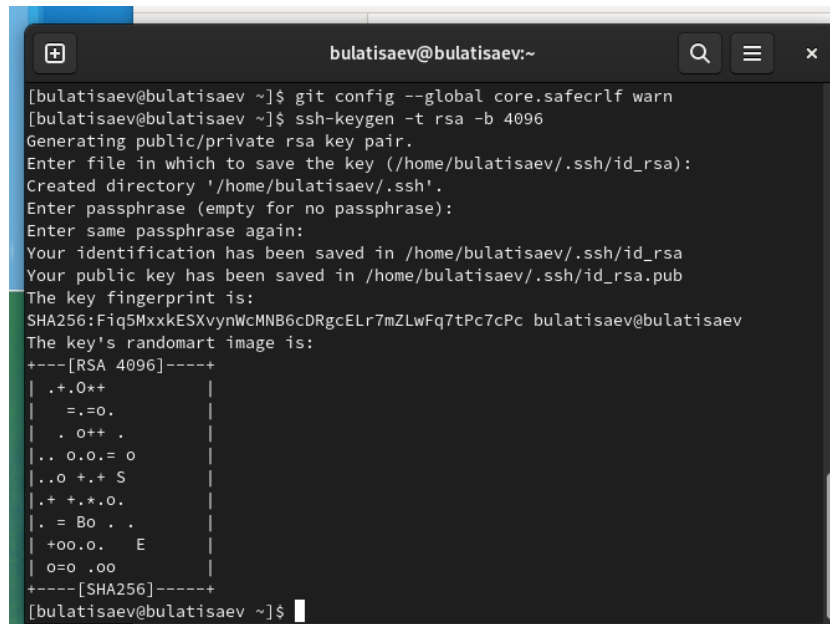
Зададим имя и email владельца репозитория, кодировку и прочие параметры.



```
[bulatisaev@bulatisaev ~]$  
[bulatisaev@bulatisaev ~]$  
[bulatisaev@bulatisaev ~]$ git config --global user.name "bulatisaev"  
[bulatisaev@bulatisaev ~]$ git config --global user.email "1132227131@pfur.ru"  
[bulatisaev@bulatisaev ~]$ git config --global core.quotepath false  
[bulatisaev@bulatisaev ~]$ git config --global init.defaultBranch master  
[bulatisaev@bulatisaev ~]$ git config --global core.autocrlf input  
[bulatisaev@bulatisaev ~]$ git config --global core.safecrlf warn  
[bulatisaev@bulatisaev ~]$
```

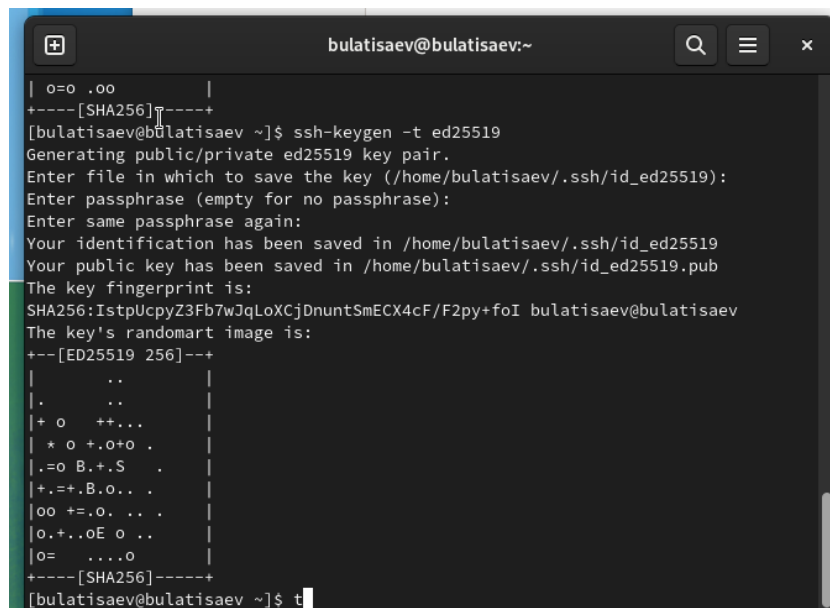
Рис. 2.2: Параметры репозитория

Создаем SSH ключи



```
bulatisaev@bulatisaev:~  
[bulatisaev@bulatisaev ~]$ git config --global core.safecrlf warn  
[bulatisaev@bulatisaev ~]$ ssh-keygen -t rsa -b 4096  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/bulatisaev/.ssh/id_rsa):  
Created directory '/home/bulatisaev/.ssh'.  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/bulatisaev/.ssh/id_rsa  
Your public key has been saved in /home/bulatisaev/.ssh/id_rsa.pub  
The key fingerprint is:  
SHA256:Fiq5MxxkESXvynWcMNB6cDRgcELr7mZLwFq7tPc7cPc bulatisaev@bulatisaev  
The key's randomart image is:  
+---[RSA 4096]-----+  
| .+.0++ |  
| =.=0. |  
| . 0++ . |  
|.. 0.0.= 0 |  
|..0 +.+ S |  
|. + +.0. |  
|. = B0 . . |  
| +00.0. E |  
| 0=0 .00 |  
+---[SHA256]-----+  
[bulatisaev@bulatisaev ~]$
```

Рис. 2.3: rsa-4096



```
bulatisaev@bulatisaev:~  
| 0=0 .00 |  
+---[SHA256]-----+  
[bulatisaev@bulatisaev ~]$ ssh-keygen -t ed25519  
Generating public/private ed25519 key pair.  
Enter file in which to save the key (/home/bulatisaev/.ssh/id_ed25519):  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/bulatisaev/.ssh/id_ed25519  
Your public key has been saved in /home/bulatisaev/.ssh/id_ed25519.pub  
The key fingerprint is:  
SHA256:IstpUcpyZ3Fb7wJqLoXCjDnuntSmECX4cF/F2py+foI bulatisaev@bulatisaev  
The key's randomart image is:  
+---[ED25519 256]---+  
| .. |  
|. .. |  
|+ 0 ++... |  
| * 0 +.0+0 . |  
|. =0 B.+ .S . |  
|+. =+.B.0.. . |  
|00 +=.0. . . |  
|0.+..0E 0 .. |  
|0= . . . .0 |  
+---[SHA256]-----+  
[bulatisaev@bulatisaev ~]$ t
```

Рис. 2.4: ed25519

Создаем GPG ключ

```
bulatisaev@bulatisaev:~  
"bulatisaev <1132227131@pfur.ru>"  
  
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 (печать  
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 (печать  
на клавиатуре, движения мыши, обращения к дискам); это даст генератору  
случайных чисел больше возможностей получить достаточное количество энтропии.  
gpg: /home/bulatisaev/.gnupg/trustdb.gpg: создана таблица доверия  
gpg: создан каталог '/home/bulatisaev/.gnupg/openpgp-revocs.d'  
gpg: сертификат отзыва записан в '/home/bulatisaev/.gnupg/openpgp-revocs.d/F140B  
F36443FC127EE7AA88B28B0B228C3A33375.rev'.  
открытый и секретный ключи созданы и подписаны.  
  
pub   rsa4096 2023-06-11 [SC]  
      F140BF36443FC127EE7AA88B28B0B228C3A33375  
uid           bulatisaev <1132227131@pfur.ru>  
sub   rsa4096 2023-06-11 [E]  
  
[bulatisaev@bulatisaev ~]$  
[bulatisaev@bulatisaev ~]$
```

Рис. 2.5: GPG ключ

Добавляем GPG ключ в аккаунт

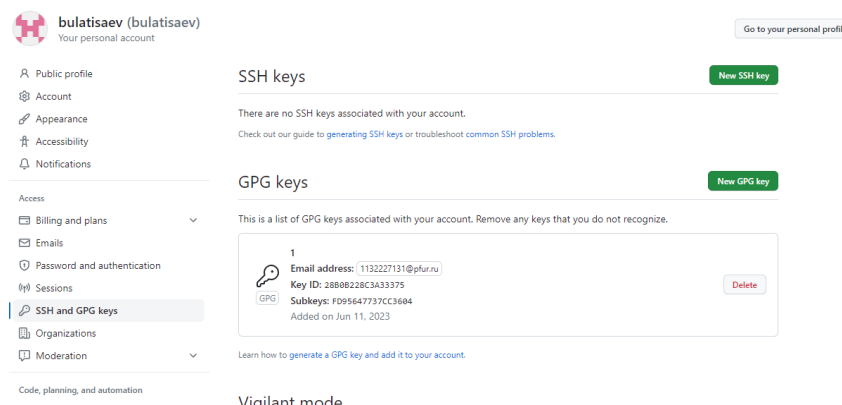


Рис. 2.6: GPG ключ

Настройка автоматических подписей коммитов git

```
bulatisaev@bulatisaev:~  
vDi9e28XJ+gRvJLKFW0jYJ7TmP6s/c8z8SaqC5MLjmrMximcVlp+PZdyqIRL1eUF  
jj+HvMt70Dq/5vne2zF/+hAaXRmuCTR6ZdMLnBpKHsc2Taf8CKoVnCO1eTYQyV4/  
n6V45dE4eA2WdSpzquErR3yEg9csDbEamVs6F+LS6LQ9CPQ6//Hg/8+P0hVTN2Uu  
Brmc4f1fNgXgooJyUthEMaiCRcdOmbi3Mch8r8g3f1WPKU9v+v/yhCYff48iQuxU  
bq10SQRlgVluiQqdSh68mG930H8KSvxcv65pnsMsl+pxkRnp5XfSrAYKVeg89z0L  
ABEBAAGJAjYEGAEIACAWIQTxQL82RD/BJ+56qIsosLIow6MzdQUCZIYXwAIbDAAK  
CRAosLIow6MzdUD5D/4rvquUkVK0ozb0ICo56FnD6Pd1VuiZEmb+rrPJLpGQruFc  
7Aa+90m0UKf1WjEeI6WwzWKq9MMjOy1PhlcrD6rm0ogMwRBirFfy1Ez9Up+h6Cp  
ZJebLB02xI7NDZ0BLi0muILRSEsgno3KUyfZ1yUzoM9yaAOBLfLPq/7BvomBWLlQ  
UL4TUFREIo+SvGyzC1b+YKAnUYBdJPaiFCFRoBPyAQ0FrM0bZW51jB/2IMfDBZSo  
J2a32Bi/qJbcNRa6w+BD9kdCLF58xcmBtmVwSVqMxqb7JdsZhMvzGPPKaYFMJKv  
3ntfy1ZTUvw/on0fowE+2RpCawXNkXnj6vbi2PzEqw2RwtqYJ94rZ6XKP0ZyopE  
eqeudTRGoRUMCj+roPdrWiAzR6vIrUqd0VF5UJsH3pEthLTLhf/VC8iZ7DoHibF  
OHTciiX9nbhrYas7xAnK4Bx/nPkf02HnW+91WvswDUSQLPRP/W33ibE00r2ztqJX  
lKKAoc+ViKqP40BBar1ct1ZqqcEvZfozTyVPH73gByzMh8K4zXi1CCQm81Wvf8Yo  
KMDoBQ+mg9n8aB99EocVfwZ17Ii0xRgN+ZIz0VFV21ffq/ujHEGd0Qu6I3iK/ZLzd  
lXRDoOhlq7y8AA0SrIdc040VpOmGfR/PXQikMFAGCgOzPHHih3l8T++NoTkH0w==  
=ZWPR  
-----END PGP PUBLIC KEY BLOCK-----  
[bulatisaev@bulatisaev ~]$ ^C  
[bulatisaev@bulatisaev ~]$ git config --global user.signingkey 28B0B228C3A33375  
[bulatisaev@bulatisaev ~]$ git config --global commit.gpgsign true  
[bulatisaev@bulatisaev ~]$ git config --global gpg.program $(which gpg2)  
[bulatisaev@bulatisaev ~]$
```

Рис. 2.7: Параметры репозитория

Настройка gh

```
bulatisaev@bulatisaev:~  
? Upload your SSH public key to your GitHub account? /home/bulatisaev/.ssh/id_rsa  
a.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: 4407-9CF0  
Press Enter to open github.com in your browser...  
[GFX1-]: glxtest: VA-API test failed: failed to initialise VAAPI connection.  
[2023-06-11T18:54:59Z ERROR glean_core::metrics::ping] Invalid reason code start  
up for ping background-update  
[GFX1-]: Unrecognized feature ACCELERATED_CANVAS2D  
✓ Authentication complete.  
- gh config set -h github.com git_protocol ssh  
✓ Configured git protocol  
✓ Uploaded the SSH key to your GitHub account: /home/bulatisaev/.ssh/id_rsa.pub  
✓ Logged in as bulatisaev  
[bulatisaev@bulatisaev ~]$ [2023-06-11T18:56:07Z ERROR viaduct::backend::ffi] Mi  
ssing HTTP status  
[2023-06-11T18:56:07Z ERROR viaduct::backend::ffi] Missing HTTP status  
[2023-06-11T18:56:07Z ERROR viaduct::backend::ffi] Missing HTTP status  
[2023-06-11T18:56:07Z ERROR viaduct::backend::ffi] Missing HTTP status  
[bulatisaev@bulatisaev ~]$
```

Рис. 2.8: Связь репозитория с аккаунтом

Загрузка шаблона репозитория и синхронизация


```
bulatisev@bulatisev:~/work/study/2022-2023/Операционны...
tation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-r
eport-template.git) зарегистрирован по пути «template/report»
Клонирование в «/home/bulatisev/work/study/2022-2023/Операционные системы/os-in
tro/template/presentation»...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Получение объектов: 100% (82/82), 92.90 КиБ | 2.44 МиБ/с, готово.
Определение изменений: 100% (28/28), готово.
Клонирование в «/home/bulatisev/work/study/2022-2023/Операционные системы/os-in
tro/template/report»...
remote: Enumerating objects: 101, done.
remote: Counting objects: 100% (101/101), done.
remote: Compressing objects: 100% (70/70), done.
remote: Total 101 (delta 40), reused 88 (delta 27), pack-reused 0
Получение объектов: 100% (101/101), 327.25 КиБ | 3.90 МиБ/с, готово.
Определение изменений: 100% (40/40), готово.
Submodule path 'template/presentation': checked out 'b1be380ee91f5809264cb755d3
16174540b753e'
Submodule path 'template/report': checked out '1d1b61dcac9c287a83917b82e3aef11a3
3b1e3b2'
[bulatisev@bulatisev Операционные системы]$
```

Рис. 2.9: Загрузка шаблона

Подготовка репозитория и коммит изменений

```
bulatisev@bulatisev:~/work/study/2022-2023/Операционны...
py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_tableno
s.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/__i
nit__.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/cor
e.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/mai
n.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/pa
ndocattributes.py
create mode 100644 project-personal/stage6/report/report.md
[bulatisev@bulatisev 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), 343.00 КиБ | 3.06 МиБ/с, готово.
Всего 37 (изменений 4), повторно использовано 0 (изменений 0), повторно использо
вано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:bulatisev/os-intro.git
385b55f..26c1fb3 master -> master
[bulatisev@bulatisev 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: