

Отчет по лабораторной работе №2

Операционные системы

Черная София Витальевна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Установка программного обеспечения	7
3.2	Базовая настройка git	8
3.3	Создание ключа SSH	8
3.4	Создание ключа GPG	9
3.5	Регистрация на Github	10
3.6	Добавление ключа GPG в Github	11
3.7	Настройка подписи Git	13
3.8	Настройка gh	13
3.9	Создание репозитория курса на основе шаблона	14
4	Выводы	17
5	Ответы на контрольные вопросы.	18
	Список литературы	21

Список иллюстраций

3.1	Установка git	7
3.2	Установка gh	7
3.3	Задаю имя и email владельца репозитория	8
3.4	Настройка utf-8 в выводе сообщений git	8
3.5	Задаю имя начальной ветке	8
3.6	Задаю параметры autoctlf и safecrlf	8
3.7	Генерация ssh ключа по алгоритму rsa	9
3.8	Генерация ssh ключа по алгоритму ed25519	9
3.9	Генерация ключа GPG	10
3.10	Аккаунт на Github	11
3.11	Вывод списка ключей	11
3.12	Копирование ключа в буфер обмена	11
3.13	Настройки GitHub	12
3.14	Добавление нового GPG ключа	12
3.15	Добавленный ключ GPG	13
3.16	Настройка автоматических подписей	13
3.17	Авторизация в gh	14
3.18	Завершение авторизации через браузер	14
3.19	Создание репозитория	15
3.20	Перемещение между директориями	15
3.21	Удаление файлов и создание каталогов	15
3.22	Отправка файлов на сервер	16

Список таблиц

1 Цель работы

Целью данной лабораторной работы - изучение идеологии и применения средств контроля версий, освоение умений по работе с git.

2 Задание

1. Создать базовую конфигурацию для работы с git
2. Создать ключ SSH
3. Создать ключ GPG
4. Настроить подписи Git
5. Зарегистрироваться на GitHub
6. Создать локальный каталон для выволнения заданий по предмет

3 Выполнение лабораторной работы

3.1 Установка программного обеспечения

Устанавливаю необходимое программное обеспечение git через терминал(рис. 3.1).

```
svchernaya@fedora:~$ sudo -i
[sudo] пароль для svchernaya:
root@fedora:~# dnf install git
Fedora 39 - x86_64 - Updates                2.5 kB/s | 19 kB      00:07
Fedora 39 - x86_64 - Updates                86 kB/s | 3.3 MB     00:38
Последняя проверка окончания срока действия метаданных: 0:00:57 назад, Пт 01 мар
2024 18:55:28.
Пакет git-2.44.0-1.fc39.x86_64 уже установлен.
Зависимости разрешены.
Нет действий для выполнения.
Выполнено!
```

Рис. 3.1: Установка git

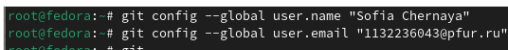
Устанавливаю необходимое программное обеспечение gh через терминал(рис. 3.2).

```
root@fedora:~# dnf install gh
Последняя проверка окончания срока действия метаданных: 0:01:38 назад, Пт 01 мар
2024 18:55:28.
Зависимости разрешены.
=====
Пакет      Архитектура  Версия      Репозиторий  Размер
=====
Установка:
gh          x86_64       2.43.1-1.fc39  updates      9.1 М
=====
Результат транзакции
=====
Установка 1 Пакет
=====
Объем загрузки: 9.1 М
Объем изменений: 46 М
Продолжить? [д/н]: д
Загрузка пакетов:
gh-2.43.1-1.fc39.x86_64.rpm                1.9 MB/s | 9.1 MB     00:04
```

Рис. 3.2: Установка gh

3.2 Базовая настройка git

Задаю в качестве имени и email владельца репозитория свои имя, фамилию и электронную почту (рис. 3.3).



```
root@fedora: # git config --global user.name "Sofia Chernaya"
root@fedora: # git config --global user.email "1132236043@pfur.ru"
```

Рис. 3.3: Задаю имя и email владельца репозитория

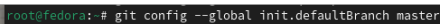
Настраиваю utf-8 в выводе сообщений git для их корректного отображения (рис. 3.4).



```
root@fedora:~# git config --global core.quotePath false
```

Рис. 3.4: Настройка utf-8 в выводе сообщений git

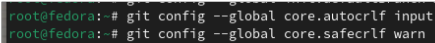
Начальной ветке задаю имя master (рис. 3.5).



```
root@fedora: # git config --global init.defaultBranch master
```

Рис. 3.5: Задаю имя начальной ветке

Задаю параметры autocrlf и safecrlf для корректного отображения конца строки(рис. 3.6).



```
root@fedora: # git config --global core.autocrlf input
root@fedora: # git config --global core.safecrlf warn
```

Рис. 3.6: Задаю параметры autocrlf и safecrlf

3.3 Создание ключа SSH

Создаю ключ ssh размером 4096 бит по алгоритму rsa(рис. 3.7).


```

root@fedora:~# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:oPGtm9tfdPi2SGocRaQgj3eq6rwHwyCbJJdvWfNuKGE root@fedora
The key's randomart image is:
+----[RSA 4096]-----+
|
| . . .
| + . .
| . o o o.
| . . + + o .
|o+ +. . S .o .
|+.o E oo . . o
| . o *o+ . .o o
| .=.o+o o+ o .
| o=+++ooo . .
+----[SHA256]-----+

```

Рис. 3.7: Генерация ssh ключа по алгоритму rsa

Создаю ключ ssh по алгоритму ed25519(рис. 3.8).

```

root@fedora:~# ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/root/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_ed25519
Your public key has been saved in /root/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:+FI9N0z746GT+0SAf5P03gKnagR0eZpCnXD2LmyiicY root@fedora
The key's randomart image is:
+---[ED25519 256]---+
|
| . o .
| = + +
| . * o. + .
| . + *. S +
| o B o+. .o
|oo + oo o=oo.
|+E .oo=B+.
| . ....=B=+.
+---[SHA256]-----+

```

Рис. 3.8: Генерация ssh ключа по алгоритму ed25519

3.4 Создание ключа GPG

Генерирую ключ GPG, затем выбираю тип ключа RSA или RSA, задаю максимальную длину ключа: 4096, оставляю неограниченный срок действия ключа. Далее отвечаю на вопросы программы о личной информации(рис. 3.9).

```

root@fedora:~# gpg --full-generate-key
gpg (GnuPG) 2.4.3; Copyright (C) 2023 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 '/root/.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

GnuPG должен составить идентификатор пользователя для идентификации ключа.
Ваше полное имя: sofia
Адрес электронной почты: 1132236043@pfur.ru

```

Рис. 3.9: Генерация ключа GPG

3.5 Регистрация на Github

У меня уже был создан аккаунт на Github, соответственно, основные данные аккаунта я так же заполняла и проводила его настройку, поэтому просто вхожу в свой аккаунт(рис. 3.10).

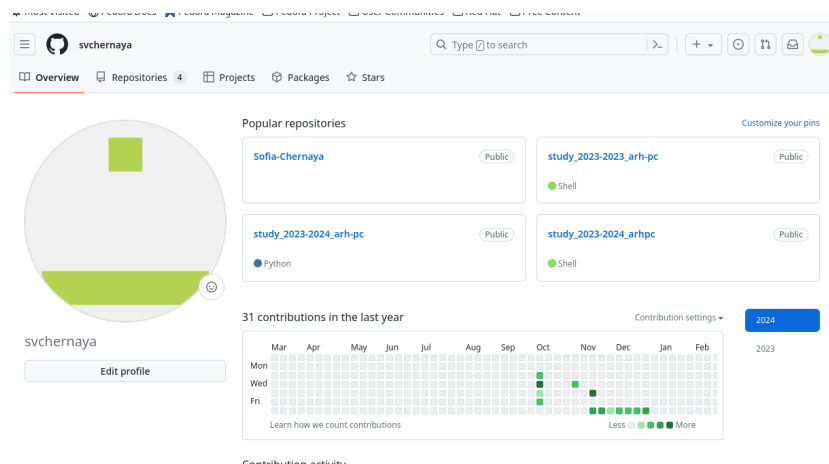


Рис. 3.10: Аккаунт на Github

3.6 Добавление ключа GPG в Github

Вывожу список созданных ключей в терминал, ищу в результате запроса отпечаток ключа (последовательность байтов для идентификации более длинного, по сравнению с самим отпечатком, ключа), он стоит после знака слеша, копирую его в буфер обмена (рис. 3.11).

```
svchernaya@fedora:~$ 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
and[keyboard]
-----
|il|sec rsa4096/929EC3B434E124AE 2024-03-01 [SC]
|ti|uid CC621F29BB19CD28B0BF63E7929EC3B434E124AE
|ssb| [ абсолютно ] SofiaChernaya <1132236043@pfur.ru>
|   |rsa4096/7379A0123C02823C 2024-03-01 [E]
```

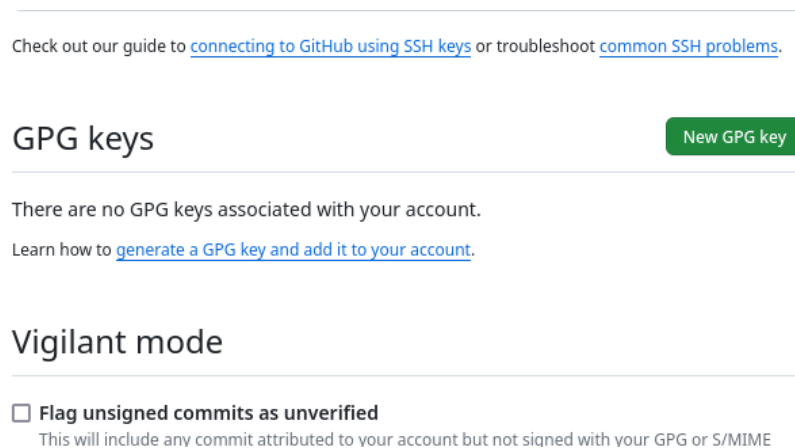
Рис. 3.11: Вывод списка ключей

Ввожу в терминале команду, с помощью которой копирую сам ключ PGP в буфер обмена, за это отвечает утилита xclip (рис. 3.12).

```
svchernaya@fedora:~$ gpg --armor --export PGP 929EC3B434E124AE | xclip -sel clip
svchernaya@fedora:~$
```

Рис. 3.12: Копирование ключа в буфер обмена

Открываю настройки Github, ищу среди них добавление PGP ключа(рис. 3.13).



Check out our guide to [connecting to GitHub using SSH keys](#) or troubleshoot [common SSH problems](#).

GPG keys

[New GPG key](#)

There are no GPG keys associated with your account.

Learn how to [generate a GPG key and add it to your account](#).

Vigilant mode

☐ **Flag unsigned commits as unverified**

This will include any commit attributed to your account but not signed with your GPG or S/MIME

Рис. 3.13: Настройки GitHub

Нажимаю на “New GPG key” и вставляю в поле ключ из буфера обмена(рис. 3.14).



Add new GPG key

Title

Key

```
jG+x1HBLUjNkvsedJKJM1eMGTInqcPxa6vtqW/D2Ilt9GqNoxyYoD8IBtDPYgSe
7JR48EyWGgWVwh7DtTPAq1x3Dk+7KV6TYkfy5BYMgmxutrXorLZ6KV5oKmhynoWh
hf/xN+pzC0w5ZNuD8sESRQ7rFcIPNpeXLq4pi9uHOI3oyECNhl38BZGNjtFHBByBI
wu56Ulga2Sck9In+VMhPL9O/vCN/UkS8yzYmFCD8FLIgPsEqexD2MWzkEAA31OZv
peqeMTv2c1wor6ZYAX5AXmQMmMLy1SjZ2Ijvnam/41bZhLkgw8Gs6YCJu2lBjKs
ng==
=jduZ
-----END PGP PUBLIC KEY BLOCK-----
```

[Add GPG key](#)


Рис. 3.14: Добавление нового GPG ключа

Я добавила ключ GPG на GitHub(рис. 3.15).

GPG keys

[New GPG key](#)

This is a list of GPG keys associated with your account. Remove any keys that you do not recognize.


GPG

Email address: 1132236043@pfur.ru
Key ID: 929EC3B434E124AE
Subkeys: 7379A0123C02823C
Added on Mar 1, 2024

[Delete](#)

Learn how to [generate a GPG key and add it to your account](#).

Рис. 3.15: Добавленный ключ GPG

3.7 Настройка подписи Git

Настраиваю автоматические подписи коммитов git: используя введенный ранее email, указываю git использовать его при создании подписей коммитов(рис. 3.16).

```
svchernaya@fedora:~  
svchernaya@fedora:~$ git config --global user.signingkey 929EC3B434E124AE  
svchernaya@fedora:~$ git config --global commit.gpgsign true  
svchernaya@fedora:~$ git config --global gpg.program $(which gpg2)  
svchernaya@fedora:~$
```

Рис. 3.16: Настройка автоматических подписей

3.8 Настройка gh

Начинаю авторизацию в gh, отвечаю на наводящие вопросы от утилиты, в конце выбираю авторизоваться через браузер(рис. 3.17).

```

svchernaya@fedora:~$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 746D-DA1F
Press Enter to open github.com in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol https
✓ Configured git protocol
✓ Logged in as svchernaya

```

Рис. 3.17: Авторизация в gh

Завершаю авторизацию на сайте(рис. 3.18).

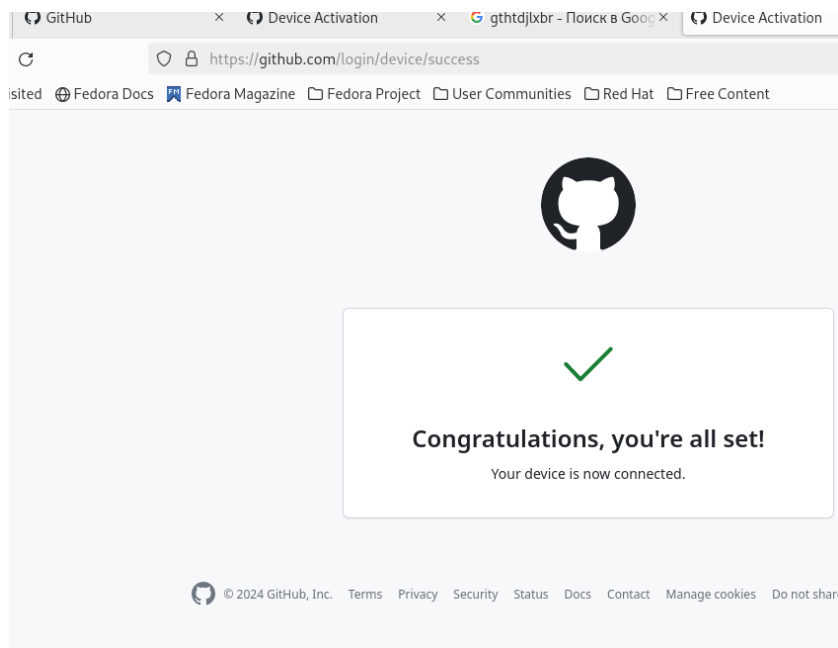


Рис. 3.18: Завершение авторизации через браузер

3.9 Создание репозитория курса на основе шаблона

Сначала создаю директорию с помощью утилиты `mkdir` и флага `-p`, который позволяет установить каталоги на всем указанном пути. После этого с помощью утилиты `cd` перехожу в только что созданную директорию “Операционные системы”. Далее в терминале ввожу команд `gh repo create study_2022-2023_os-intro --template yamadharma/course-directory-student-trmplate --public`, чтобы создать

репозиторий на основе шаблона репозитория. После этого клонирую репозиторий к себе в директорию, я указываю ссылку с протоколом https, а не ssh, потому что при авторизации в gh выбрала протокол https(рис. 3.19).

```
svchernaya@fedora: /work/study/2023-2024/Операционные системы$ git clone --recursive https://github.com/svchernaya/study_2023-2024_os-intro.git os-intro
Клонирование в «os-intro»...
remote: Enumerating objects: 32, done.
remote: Counting objects: 100% (32/32), done.
remote: Compressing objects: 100% (31/31), done.
remote: Total 32 (delta 1), reused 18 (delta 0), pack-reused 0
Получение объектов: 100% (32/32), 18.60 KiB | 442.00 KiB/c, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пути «tem
plate/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистрирован по пути «template/repo
rt»
Клонирование в «/home/svchernaya/work/study/2023-2024/Операционные системы/os-intro/template/presentation»...
remote: Enumerating objects: 95, done.
remote: Counting objects: 100% (95/95), done.
remote: Compressing objects: 100% (67/67), done.
remote: Total 95 (delta 34), reused 87 (delta 26), pack-reused 0
Архитектура компьютеров и
git add .
```

Рис. 3.19: Создание репозитория

Перехожу в каталог курса с помощью утилиты cd, проверяю содержание каталога с помощью утилиты ls(рис. 3.20).

```
svchernaya@fedora: /work/study/2023-2024/Операционные системы/os-intro$ ls
CHANGELOG.md  config  COURSE  LICENSE  Makefile  package.json  README.en.md  README.git-flow.md  README.md  template
svchernaya@fedora: /work/study/2023-2024/Операционные системы/os-intro$
```

Рис. 3.20: Перемещение между директориями

Удаляю лишние файлы с помощью утилиты rm, далее создаю необходимые каталоги используя makefile (рис. 3.21).

```
svchernaya@fedora: /work/study/2023-2024/Операционные системы/os-intro$ rm package.json
svchernaya@fedora: /work/study/2023-2024/Операционные системы/os-intro$ echo os-intro > COURSE-
svchernaya@fedora: /work/study/2023-2024/Операционные системы/os-intro$ make
Usage:
make <target>

Targets:
list           List of courses
prepare       Generate directories structure
submodule     Update submodules
```

Рис. 3.21: Удаление файлов и создание каталогов

Добавляю все новые файлы для отправки на сервер (сохраняю добавленные изменения) с помощью команды git add и комментирую их с помощью git commit. Отправляю файлы на сервер с помощью git push (рис. fig. 3.22).

```
svchernaya@fedora:~/work/study/2023-2024/Операционные системы/os-intro$ git commit -am 'feat(main): make course structure'
[master 66a0aed] feat(main): make course structure
1 file changed, 14 deletions(-)
delete mode 100644 package.json
svchernaya@fedora:~/work/study/2023-2024/Операционные системы/os-intro$ git push
Перечисление объектов: 3, готово.
Подсчет объектов: 100% (3/3), готово.
Сжатие объектов: 100% (2/2), готово.
Запись объектов: 100% (2/2), 985 байтов | 985.00 Кб/с, готово.
Total 2 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/svchernaya/study_2023-2024_os-intro.git
23a8ec8..66a0aed master -> master
svchernaya@fedora:~/work/study/2023-2024/Операционные системы/os-intro$
```

Рис. 3.22: Отправка файлов на сервер

4 Выводы

При выполнении данной лабораторной работы я изучила идеологию и применение средств контроля версий, освоила умение по работе с git.

5 Ответы на контрольные вопросы.

1. Системы контроля версий (VCS) - программное обеспечение для облегчения работы с изменяющейся информацией. Они позволяют хранить несколько версий изменяющейся информации, одного и того же документа, может предоставить доступ к более ранним версиям документа. Используется для работы нескольких человек над проектом, позволяет посмотреть, кто и когда внес какое-либо изменение и т. д. VCS применяются для: Хранения полной истории изменений, сохранения причин всех изменений, поиска причин изменений и совершивших изменение, совместной работы над проектами.
2. Хранилище – репозиторий, хранилище версий, в нем хранятся все документы, включая историю их изменения и прочей служебной информацией. commit – отслеживание изменений, сохраняет разницу в изменениях. История – хранит все изменения в проекте и позволяет при необходимости вернуться/обратиться к нужным данным. Рабочая копия – копия проекта, основанная на версии из хранилища, чаще всего последней версии.
3. Централизованные VCS (например: CVS, TFS, AccuRev) – одно основное хранилище всего проекта. Каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет, затем добавляет изменения обратно в хранилище. Децентрализованные VCS (например: Git, Bazaar) – у каждого пользователя свой вариант репозитория (возможно несколько вариантов), есть возможность добавлять и забирать изменения из любого

репозитория. В отличие от классических, в распределенных (децентрализованных) системах контроля версий центральный репозиторий не является обязательным.

4. Сначала создается и подключается удаленный репозиторий, затем по мере изменения проекта эти изменения отправляются на сервер.
5. Участник проекта перед началом работы получает нужную ему версию проекта в хранилище, с помощью определенных команд, после внесения изменений пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются. К ним можно вернуться в любой момент.
6. Хранение информации о всех изменениях в вашем коде, обеспечение удобства командной работы над кодом.
7. Создание основного дерева репозитория: `git init`

Получение обновлений (изменений) текущего дерева из центрального репозитория: `git pull`

Отправка всех произведённых изменений локального дерева в центральный репозиторий: `git push`

Просмотр списка изменённых файлов в текущей директории: `git status` Просмотр текущих изменений: `git diff`

Сохранение текущих изменений: добавить все изменённые и/или созданные файлы и/или каталоги: `git add` .

добавить конкретные изменённые и/или созданные файлы и/или каталоги: `git add имена_файлов`

удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории): `git rm имена_файлов`

Сохранение добавленных изменений:

сохранить все добавленные изменения и все изменённые файлы: `git commit -am 'Описание коммита'`

сохранить добавленные изменения с внесением комментария через встроенный редактор: `git commit`

создание новой ветки, базирующейся на текущей: `git checkout -b имя_ветки`

переключение на некоторую ветку: `git checkout имя_ветки` (при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой)

отправка изменений конкретной ветки в центральный репозиторий: `git push origin имя_ветки`

слияние ветки с текущим деревом: `git merge --no-ff имя_ветки`

Удаление ветки:

удаление локальной уже слитой с основным деревом ветки: `git branch -d имя_ветки`

принудительное удаление локальной ветки: `git branch -D имя_ветки`

удаление ветки с центрального репозитория: `git push origin :имя_ветки`

8. `git push -all` отправляем из локального репозитория все сохраненные изменения в центральный репозиторий, предварительно создав локальный репозиторий и сделав предварительную конфигурацию.
9. Ветвление - один из параллельных участков в одном хранилище, исходящих из одной версии, обычно есть главная ветка. Между ветками, т. е. их концами возможно их слияние. Используются для разработки новых функций.
10. Во время работы над проектом могут создаваться файлы, которые не следуют добавлять в репозиторий. Например, временные файлы. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл `.gitignore` с помощью сервисов.

Список литературы

<https://esystem.rudn.ru/mod/page/view.php?id=1098790>