

Отчет по лабораторной работе №2

Операционные системы

Мориссала Донзо НКАБд-01-24

Содержание

Цель работы.....	1
Задание.....	1
Выполнение лабораторной работы	2
Установка программного обеспечения	2
Базовая настройка git.....	2
Создание ключа SSH.....	3
Создание ключа GPG.....	3
Регистрация на Github	4
Добавление ключа GPG в Github	5
Настроить подписи Git	6
Настройка gh.....	6
Создание репозитория курса на основе шаблона	7
Выводы.....	9
Ответы на контрольные вопросы.....	9
Список литературы.....	11

Цель работы

Цель данной лабораторной работы – изучение идеологии и применения средств контроля версий, освоение умения по работе с git.

Задание

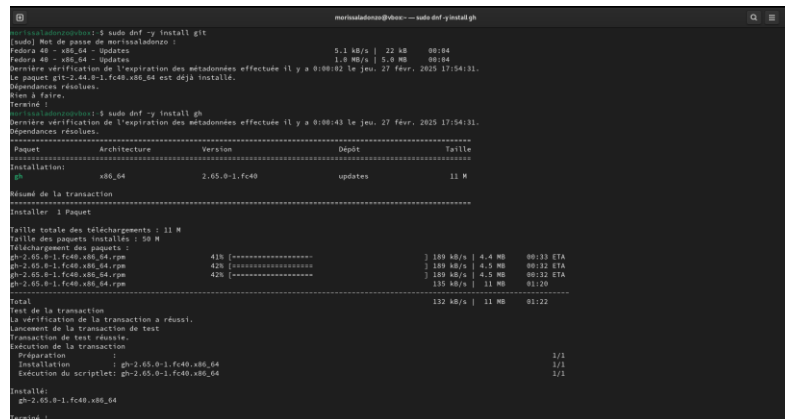
1. Создать базовую конфигурацию для работы с git
2. Создать ключ SSH
3. Создать ключ GPG
4. Настроить подписи Git

5. Зарегистрироваться на GitHub
6. Создать локальный каталог для выполнения заданий по предмету.

Выполнение лабораторной работы

Установка программного обеспечения

Устанавливаю необходимое программное обеспечение git и gh через терминал с помощью команд: `dnf install git` и `dnf install gh` (рис. @fig:001).



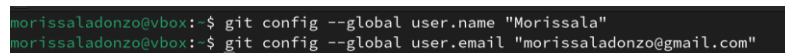
```
morissaladonzo@vbox: ~$ sudo dnf -y install git
[sudo] Mot de passe de morissaladonzo : 
Fedora 40 - x86_64 - updates                               5.1 kB/s | 22 kB   00:04
Fedora 40 - x86_64 - updates                               1.6 MB/s | 5.9 MB   00:04
 Dernière vérification de l'expiration des métadonnées effectuée il y a 0:00:02 le jeu. 27 févr. 2025 17:54:31.
 Le paquet git-2.44.0-1.fc40.x86_64 est déjà installé.
 Dépendances résolues.
 Rien à faire.
 Terminé !

morissaladonzo@vbox: ~$ sudo dnf -y install gh
 Dernière vérification de l'expiration des métadonnées effectuée il y a 0:00:43 le jeu. 27 févr. 2025 17:54:31.
 Dépendances résolues.
=====
 Paquet      Architecture  Version      Dépôt      Taille
=====
 Installation:
 gh          x86_64        2.65.0-1.fc40 updates     11 M
=====
 Résolu de la transaction
=====
 Installer 1 Paquet
=====
 Taille totale des téléchargements : 11 M
 Taille des paquets installés : 30 M
 Téléchargement des paquets :
 gh-2.65.0-1.fc40.x86_64.rpm      410 [=====]      1100 kB/s | 4.4 MB   00:13 ETA
 gh-2.65.0-1.fc40.x86_64.rpm      420 [=====]      1100 kB/s | 4.5 MB   00:12 ETA
 gh-2.65.0-1.fc40.x86_64.rpm      420 [=====]      1100 kB/s | 4.5 MB   00:12 ETA
 gh-2.65.0-1.fc40.x86_64.rpm      420 [=====]      1100 kB/s | 4.5 MB   00:12 ETA
 Total: 132 kB/s | 11 MB   01:22
 Test de la transaction
 Vérification de la transaction a réussi.
 Lancement de la transaction de test
 Lancement de test réussi.
 Exécution de la transaction
 Préparation : 1/1
 Installation : gh-2.65.0-1.fc40.x86_64 1/1
 Exécution du scriptlet: gh-2.65.0-1.fc40.x86_64 1/1
 Installés :
 gh-2.65.0-1.fc40.x86_64
 Terminé !
```

Установка git и gh

Базовая настройка git

Задаю в качестве имени и email владельца репозитория свои имя, фамилию и электронную почту (рис. @fig:002).



```
morissaladonzo@vbox: ~$ git config --global user.name "Morissala"
morissaladonzo@vbox: ~$ git config --global user.email "morissaladonzo@gmail.com"
```

Задаю имя и email владельца репозитория

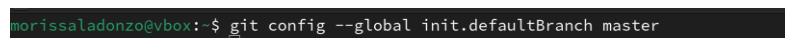
Настраиваю utf-8 в выводе сообщений git для их корректного отображения (рис. @fig:003).



```
morissaladonzo@vbox: ~$ git config --global core.quotePath false
morissaladonzo@vbox: ~$
```

Настройка utf-8 в выводе сообщений git

Начальной ветке задаю имя master (рис. @fig:004).



```
morissaladonzo@vbox: ~$ git config --global init.defaultBranch master
```

Задаю имя начальной ветки

Задаю параметры autocrlf и safecrlf для корректного отображения конца строки (рис. @fig:005).

```
morissaladonzo@vbox:~$ git config --global core.autocrlf input
morissaladonzo@vbox:~$ git config --global core.safecrlf warn
```

Задаю параметры autocrlf и safecrlf

Создание ключа SSH

Создаю ключ ssh размером 4096 бит по алгоритму rsa (рис. @fig:006).

```
morissaladonzo@vbox:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/morissaladonzo/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/morissaladonzo/.ssh/id_rsa
Your public key has been saved in /home/morissaladonzo/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:n4x4S5cJ8xWLUaVlUrlQLe9lFnCXJkqM9LXc0StwgEU morissaladonzo@vbox
The key's randomart image is:
+---[RSA 4096]-----+
|
|..*E**B=+|
|o.=+B*+=|
|o.**o=o|
|+ +.|=|
|S . o .+.|
|. B = .|
|. + 0|
|o o|
|. |
+---[SHA256]-----+
```

Генерация ssh ключа по алгоритму rsa

Создаю ключ ssh по алгоритму ed25519 (рис. @fig:007).

```
morissaladonzo@vbox:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/morissaladonzo/.ssh/id_ed25519):
/home/morissaladonzo/.ssh/id_ed25519 already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/morissaladonzo/.ssh/id_ed25519
Your public key has been saved in /home/morissaladonzo/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:K/WKR9Ru+Wp9oC9rZ75UWjztemcEgPqFj3VIw52Cgwo morissaladonzo@vbox
The key's randomart image is:
+--[ED25519 256]--+
|
|. + . . |
|E . + * o |
|. . o + = |
|. o o +.o. |
|.So * .=. .|
|.o* o+ o. |
|.....+o. .|
|o..=,= .+|
|...oo0+o.o.|
+---[SHA256]-----+
```

Генерация ssh ключа по алгоритму ed25519

Создание ключа GPG

Генерирую ключ GPG, затем выбираю тип ключа RSA and RSA, задаю максимальную длину ключа: 4096, оставляю неограниченный срок действия ключа. Далее отвечаю на вопросы программы о личной информации (рис. @fig:008).

```

morissaladonzo@vbox:~$ gpg --full-generate-key
gpg (GnuPG) 2.4.4; 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: répertoire « /home/morissaladonzo/.gnupg » créé
Sélectionnez le type de clef désiré :
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) *default*
  (10) ECC (signature seule)
  (14) Existing key from card
Quel est votre choix ? 1
les clefs RSA peuvent faire une taille comprise entre 1024 et 4096 bits.
Quelle taille de clef désirez-vous ? (3072) 4096
La taille demandée est 4096 bits
Veuillez indiquer le temps pendant lequel cette clef devrait être valable.
  0 = la clef n'expire pas
  <n> = la clef expire dans n jours
  <n>w = la clef expire dans n semaines
  <n>m = la clef expire dans n mois
  <n>y = la clef expire dans n ans
Pendant combien de temps la clef est-elle valable ? (0) 0
La clef n'expire pas du tout
Est-ce correct ? (o/N) o

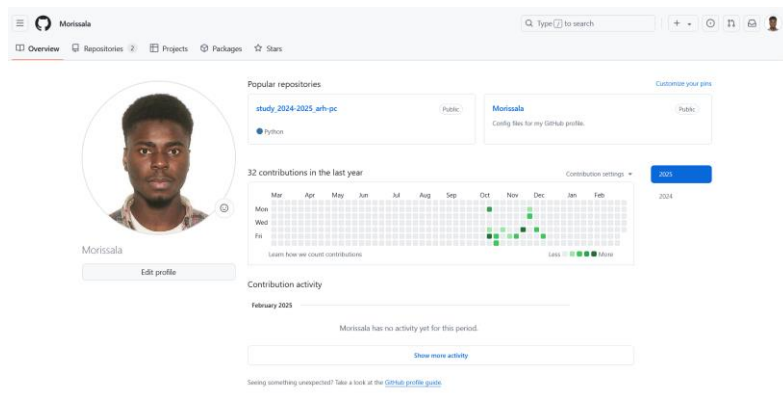
GnuPG doit construire une identité pour identifier la clef.

Nom réel : DONZO
Adresse électronique : morissaladonzo@gmail.com
Commentaire : je suis fans des maths

```

Генерация ключа

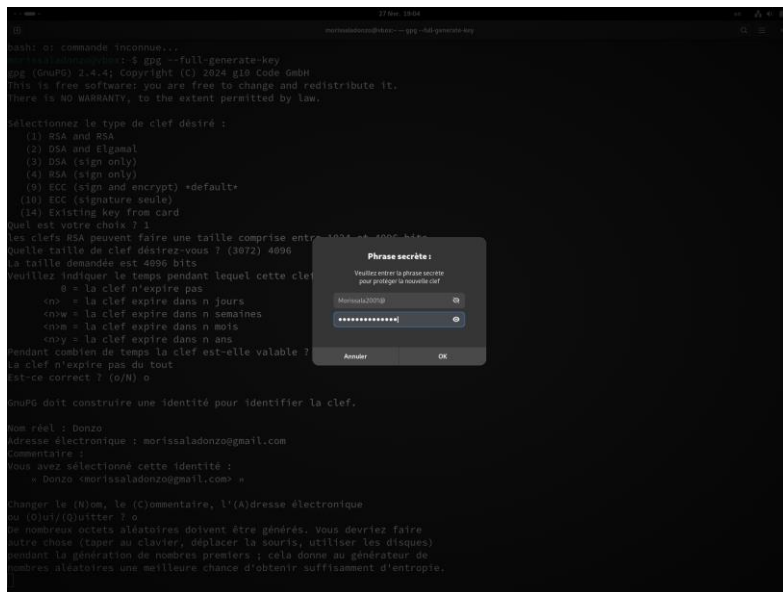
Ввожу фразу-пароль для защиты нового ключа (рис. @fig:009).



Защита ключа GPG

Регистрация на Github

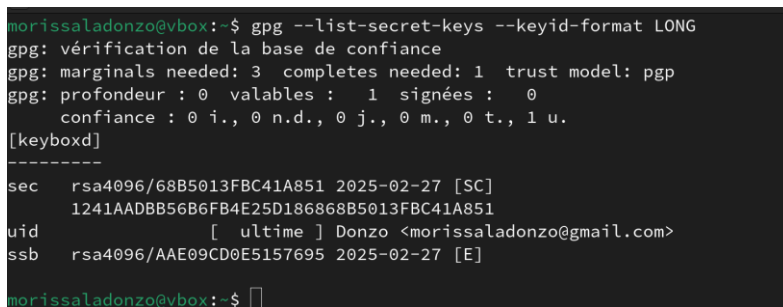
У меня уже был создан аккаунт на Github, соответственно, основные данные аккаунта я так же заполняла и проводила его настройку, поэтому просто вхожу в свой аккаунт (рис. @fig:010).



Аккаунт на Github

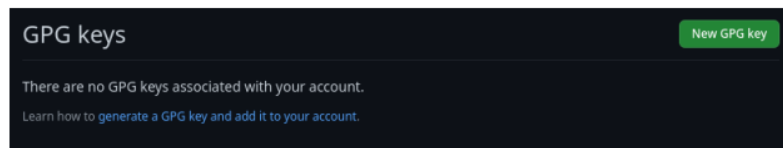
Добавление ключа GPG в Github

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 следа, копирую его в буфер обмена (рис. @fig:011).



Вывод списка ключей

Ввожу в терминале команду, с помощью которой копирую сам ключ GPG в буфер обмена, за это отвечает утилита xclip (рис. @fig:012).



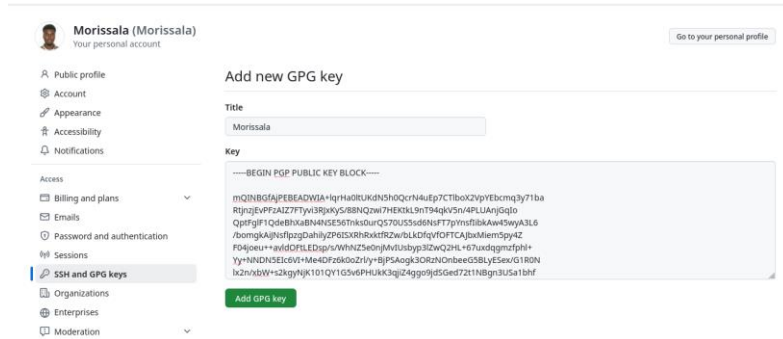
Копирование ключа в буфер обмена

Открываю настройки GitHub, ищу среди них добавление GPG ключа (рис. @fig:013).

```
morissaladonzo@vbox:~$ gpg --armor --export 68B5013FBC41A851 | xclip -sel clip
morissaladonzo@vbox:~$
```

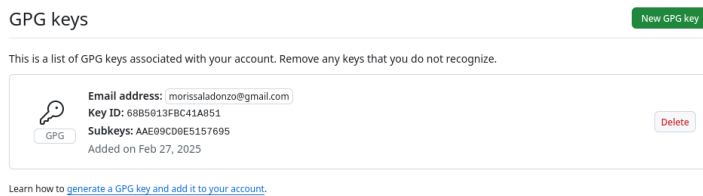
Настройка GitHub

Нажимаю на “New GPG key” и вставляю в поле ключ из буфера обмена (рис. @fig:014).



Добавление нового PGP ключа

Я добавила ключ GPG на GitHub (рис. @fig:015).



Добавленный ключ GPG

Настроить подписи Git

Настраиваю автоматические подписи коммитов git: используя введенный ранее email, указываю git использовать его при создании подписей коммитов (рис. @fig:016).

```
morissaladonzo@vbox:~$ git config --global user.signingkey 68B5013FBC41A851
morissaladonzo@vbox:~$ git config --global commit.gpgsign true
morissaladonzo@vbox:~$ git config --global gpg.program $(which gpg2)
morissaladonzo@vbox:~$
```

Настройка подписей Git

Настройка gh

Начинаю авторизацию в gh, отвечаю на наводящие вопросы от утилиты, в конце выбираю авторизоваться через браузер (рис. @fig:017).

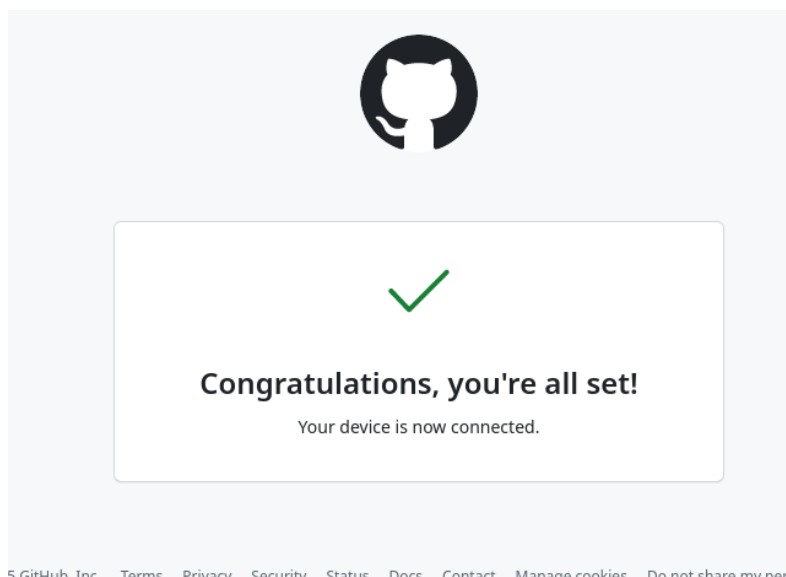
```

morissaladonzo@vbox:~$ gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? [Use arrows to move, type to filter]
? 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

```

Авторизация в gh

Завершаю авторизацию на сайте (рис. @fig:018).



Завершение авторизации через браузер

Вижу сообщение о завершении авторизации под именем evdvorkina (рис. @fig:019).

```

✓ Authentication complete.
- gh config set -h github.com git_protocol https
✓ Configured git protocol
✓ Logged in as Morissala

```

Завершение авторизации

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 команду `gh repo create study_2022-2023_os-intro -template uamadharm/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. @fig:020).

```

morissaladonzo@vbox: /work/study/2024-2025/Операционные системы$ git clone --recursive https://github.com/Morissala/study_2024-2025_os-intro.git os-intro
Clonage dans 'os-intro'...
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)
Réception d'objets: 100% (36/36), 19.37 Kio | 431.90 Kio/s, fait.
Résolution des deltas: 100% (1/1), fait.
Sous-module 'template/presentation' (https://github.com/yamadharma/academic-presentation-markdown-template.git) enregistré pour le chemin 'template/presentation'
Sous-module 'template/report' (https://github.com/yamadharma/academic-laboratory-report-template.git) enregistré pour le chemin 'template/report'
Clonage dans '/home/morissaladonzo/work/study/2024-2025/Операционные системы/os-intro/template/presentation'...
remote: Enumerating objects: 111, done.
remote: Counting objects: 100% (111/111), done.
remote: Compressing objects: 100% (97/97), done.
remote: Total 111 (delta 42), reused 100 (delta 31), pack-reused 0 (from 0)
Réception d'objets: 100% (111/111), 102.17 Kio | 1.02 Mio/s, fait.
Résolution des deltas: 100% (42/42), fait.
Clonage dans '/home/morissaladonzo/work/study/2024-2025/Операционные системы/os-intro/template/report'...
remote: Enumerating objects: 142, done.
remote: Counting objects: 100% (142/142), done.
remote: Compressing objects: 100% (97/97), done.
remote: Total 142 (delta 60), reused 121 (delta 39), pack-reused 0 (from 0)
Réception d'objets: 100% (142/142), 341.09 Kio | 1.54 Mio/s, fait.
Résolution des deltas: 100% (60/60), fait.
Chemin de sous-module 'template/presentation' : 'c9b2712b4b2d431ad5866c9c72a02bd2fcald4a6' extrait
Chemin de sous-module 'template/report' : 'c2d022effe7b3e493702ef9561ab185fc748' extrait
morissaladonzo@vbox: /work/study/2024-2025/Операционные системы$

```

Создание репозитория

Перехожу в каталог курса с помощью утилиты `cd`, проверяю содержание каталога с помощью утилиты `ls` (рис. @fig:021).

```

morissaladonzo@vbox: /work/study/2024-2025/Операционные системы$ cd os-intro
morissaladonzo@vbox: /work/study/2024-2025/Операционные системы/os-intro$ ls
CHANGELOG.md  config  COURSE  LICENSE  Makefile  package.json  README.en.md  README.git-flow.md  README.md  template

```

Перемещение между директориями

Удаляю лишние файлы с помощью утилиты `rm`, далее создаю необходимые каталоги используя `makefile` (рис. @fig:022).

```

morissaladonzo@vbox: /work/study/2024-2025/Операционные системы/os-intro$ rm package.json
morissaladonzo@vbox: /work/study/2024-2025/Операционные системы/os-intro$ echo os-intro > COURSE
morissaladonzo@vbox: /work/study/2024-2025/Операционные системы/os-intro$ make
Usage:
  make <target>

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

```

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` (рис. @fig:023).

```

morissaladonzo@vbox: /work/study/2024-2025/Операционные системы/os-intro$ git add .
morissaladonzo@vbox: /work/study/2024-2025/Операционные системы/os-intro$ git commit -am 'feat(main): make course structure'
[master 7733239] feat(main): make course structure
 2 files changed, 1 insertion(+), 14 deletions(-)
 delete mode 100644 package.json
morissaladonzo@vbox: /work/study/2024-2025/Операционные системы/os-intro$

```

Отправка файлов на сервер

Отправляю файлы на сервер с помощью `git push` (рис. @fig:024).

```

morissaladonzo@vbox: /work/study/2024-2025/Операционные системы/os-intro$ git push
Énumération des objets: 5, fait.
Décompte des objets: 100% (5/5), fait.
Compression par delta en utilisant jusqu'à 3 fils d'exécution
Compression des objets: 100% (2/2), fait.
Écriture des objets: 100% (3/3), 945 octets | 945.00 Kio/s, fait.
Total 3 (delta 1), réutilisés 0 (delta 0), réutilisés du paquet 0 (depuis 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Morissala/study_2024-2025_os-intro.git
  afa90f0..7733239  master -> master
morissaladonzo@vbox: /work/study/2024-2025/Операционные системы/os-intro$

```

Отправка файлов на сервер

Выводы

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

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 понлои 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

1. Лабораторная работа № 2 [Электронный ресурс] URL: <https://esystem.rudn.ru/mod/page/view.php?id=970819>