Oтчёт по лабораторной работе № 2

Первоначальна настройка git.

Сергей Витальевич Павлюченков

Содержание

# 1 Цель работы

Изучить идеологию и применение средств контроля версий.

Освоить умения по работе с git.

# 2 Задание

Создать базовую конфигурацию для работы с git.

Создать ключ SSH.

Создать ключ PGP.

Настроить подписи git.

Зарегистрироваться на Github.

Создать локальный каталог для выполнения заданий по предмету.

# 3 Выполнение лабораторной работы

Устанавливаю git командой dnf install git.

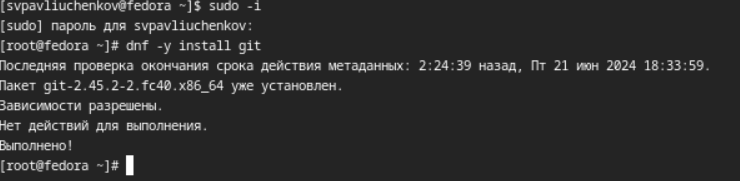


Рис. 1: Установка git

Устанавливаю gh командой dnf install gh.

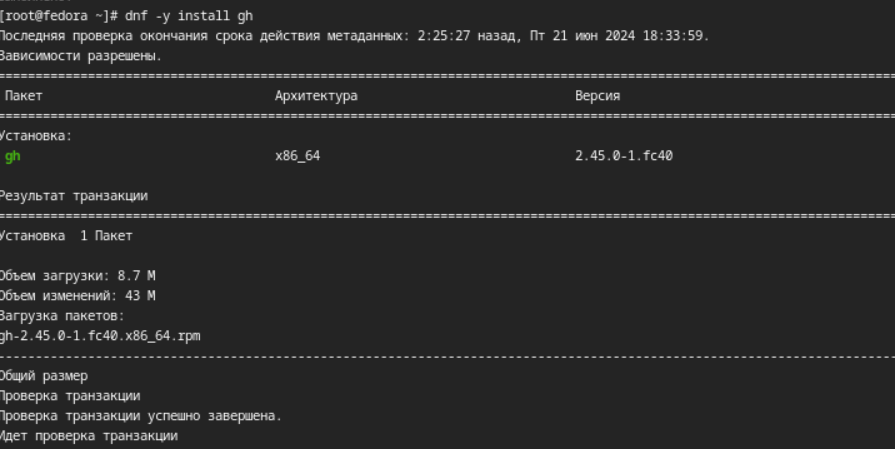


Рис. 2: Установка gh

Задаю имя и email владельца репозитория, настраиваю utf-8 в выводе сообщений git, задаю имя начальной ветки и меняю параметры autocrlf и safecrlf.(рис. 1).

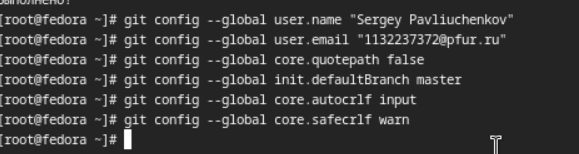


Рис. 3: Использование git config

Создаю ключи ssh по двум алгоритмам - по алгоритму rsa с ключём размером 4096 бит и по алгоритму ed25519(рис. 4).

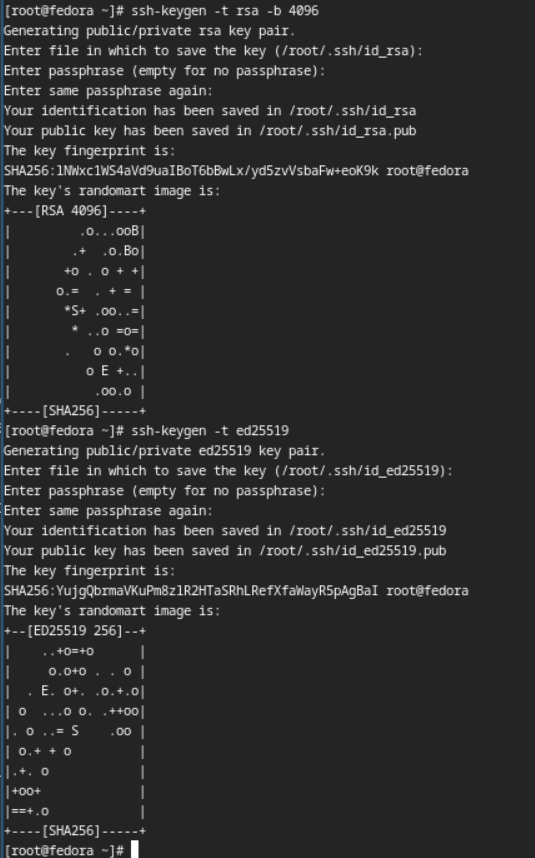


Рис. 4: Создание SSH-ключа

Генерирую pgp ключи, используя команду gpg –full-generate-key (рис. 5).

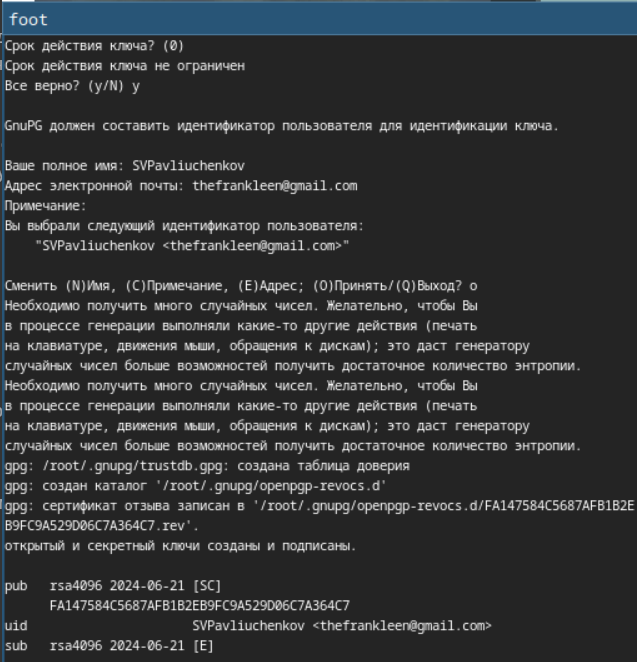


Рис. 5: Создание PGP-ключа

Также выбираю опции, а именно: тип RSA and RSA; размер 4096;срок действия не истекает никогда

Получаю такие ключи.

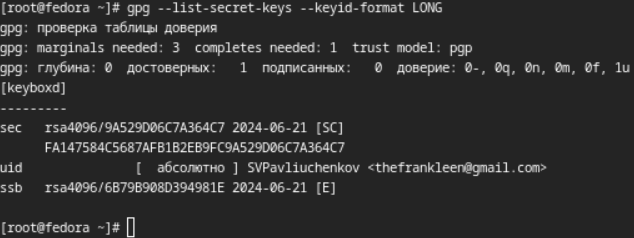


Рис. 6: Полученные PGP-ключи

Копирую отпечаток ключа, и захожу на github, чтобы добавить новый ключ.

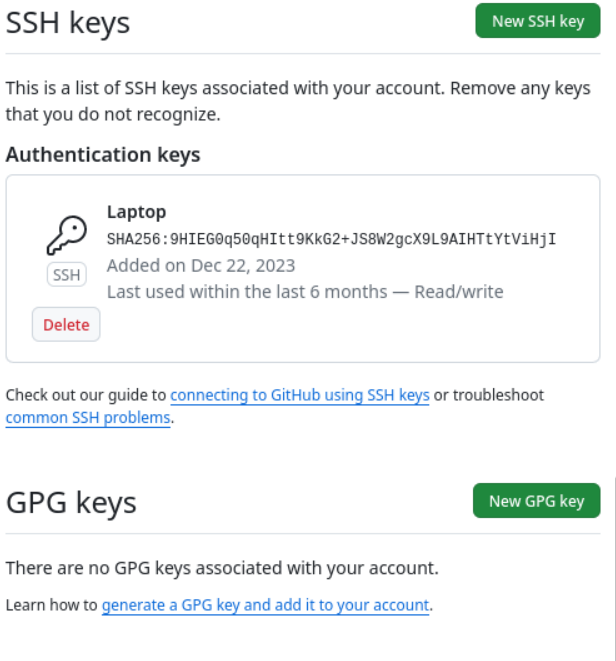


Рис. 7: Настройки github

Вношу отпечаток своего ключа на сайт.

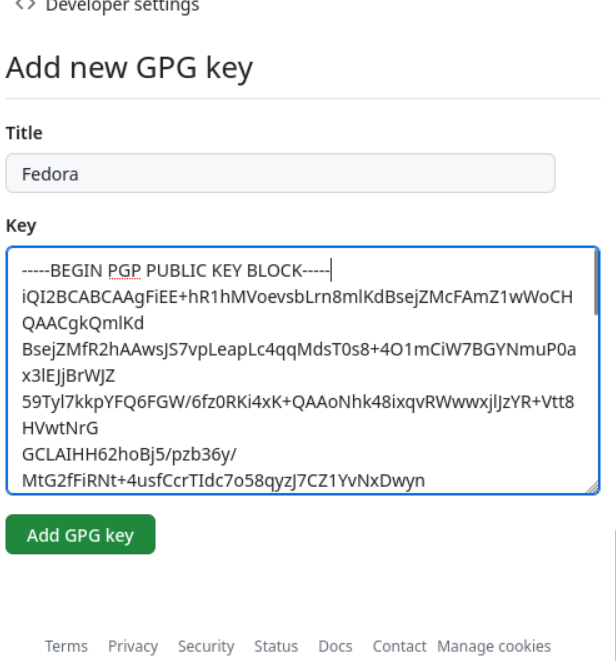


Рис. 8: Заполнение ключа

У меня получилось добавить новый ключ.

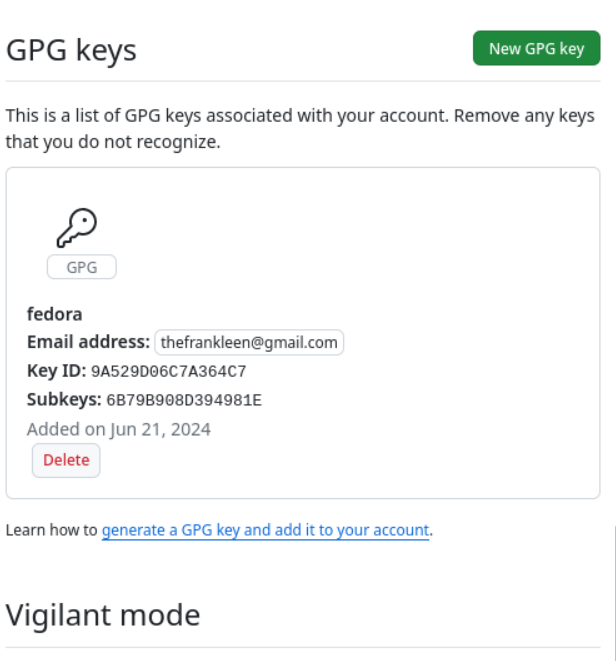


Рис. 9: Новый ключ

Указываю Git применять введенную почту при подписи коммитов.

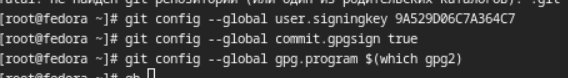


Рис. 10: Настраиваю подписи коммитов git.

Авторизуюсь используя gh auth login

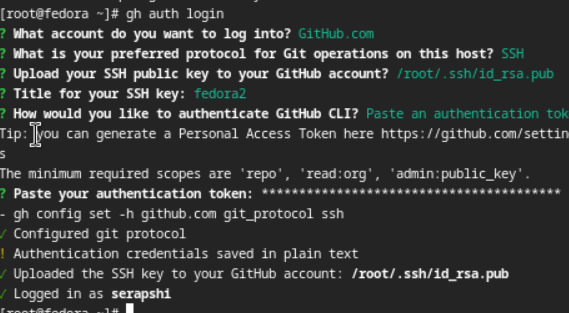


Рис. 11: Авторизация gh

Создаю репозиторий курса на основе шаблона

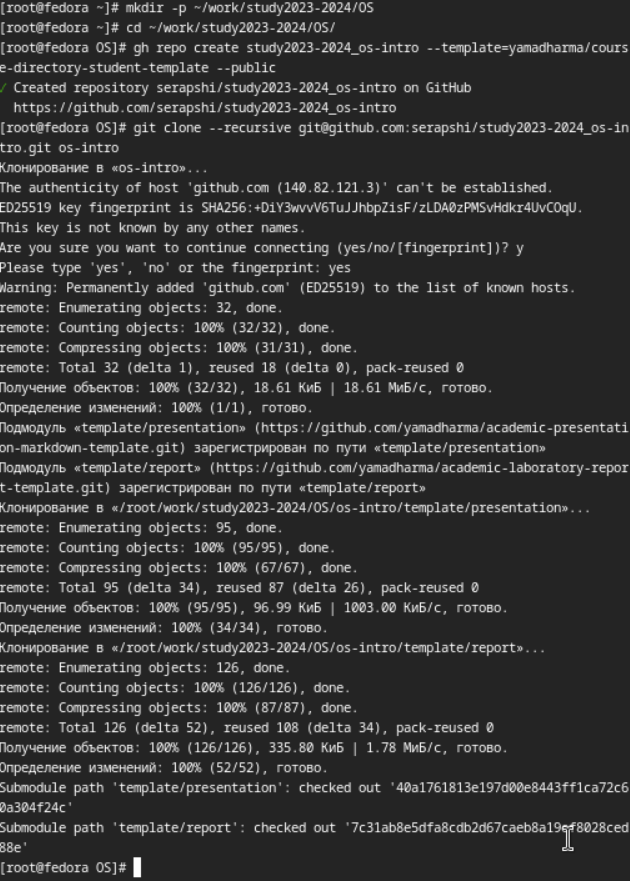


Рис. 12: Создание репозитория курса на основе шаблона

Удаляю лишние файлы(рис. 13).

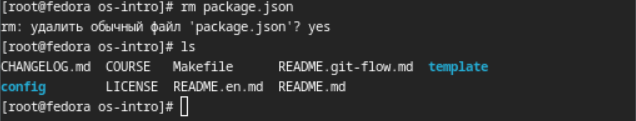


Рис. 13: Настройка каталога курса

Создаю необходимые каталоги.

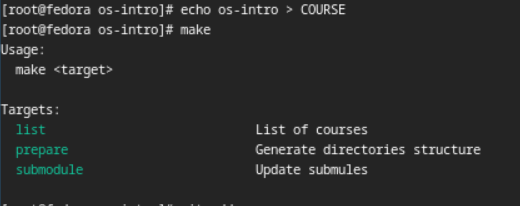


Рис. 14: Создание необходимых каталогов

Отправляю файлы на сервер.

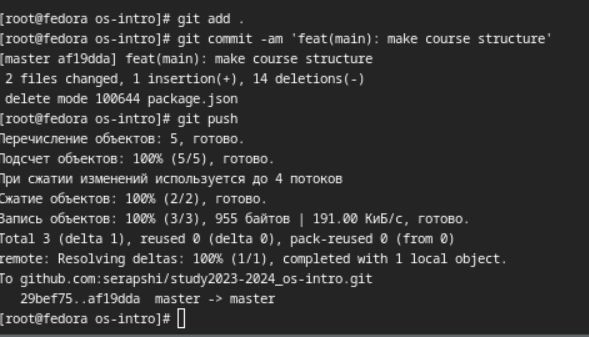


Рис. 15: Использование

# 4 Выводы

Я подготовил git репозиторий для последующей работы по этому курсу и освежил навыки работы по работе с git.

# 5 Контрольные вопросы

Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются?

Это ПО для отслеживания изменений в файлах. Оно может быть использовано, как в разработке больших систем, так и небольших программ. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.

Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.

Централизованные хранят все данные на одном сервере, как Яндекс. А децентрализованные такие, как git хранят часть данных на сервере, а другие у самих разработчиков.

Опишите действия с VCS при единоличной работе с хранилищем.

После внесения изменений, пользователь размещает новую версию в хранилище.

Опишите порядок работы с общим хранилищем VCS.

Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил.

Каковы основные задачи, решаемые инструментальным средством git?

Ветвление, слияние и управление репозиторием.

Назовите и дайте краткую характеристику командам git.

git add - направляет элемент к загрузке на сервер, git push - загружает элементы на сервер

Приведите примеры использования при работе с локальным и удалённым репозиториями.

При работе с локальным: git add .; git commit ‘feat(main): commit’; git push. При работе с удаленным: git clone –reversive (URL).

Что такое и зачем могут быть нужны ветви (branches)?

Branch(ветка) - это ответвление, от целого репозитория, где можно заниматься разработкой отпределенных функций. Для удобства разработки и/или исправения ошибок.

Как и зачем можно игнорировать некоторые файлы при commit?

Во время работы над проектом так или иначе могут создаваться файлы, которые не требуется добавлять в последствии в репозиторий. Например, временные файлы, создаваемые редакторами, или объектные файлы, создаваемые компиляторами. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл .gitignore с помощью сервисов