Лабораторная работа №2

Дисциплина: Операционные системы

Алиева Милена Арифовна

Содержание

# 1 Цель работы

Целью работы является изучить идеологию и применение средств контроляверсий git. Приобрести практические навыки по работе с системой git.

# 2 Задание

1. Базовая настройка git
2. Создание SSH ключа
3. Создание GPG ключа
4. Создание рабочего пространства и репозитория курса на основе шаблона
5. Настройка каталога курса

# 3 Теоретическое введение

Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями

Также перечислим наиболее часто используемые команды git.

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 # Выполнение лабораторной работы

1. Сначала сделаем предварительную конфигурацию git. Откроем терминал и введём следующие команды, указав имя и email владельца репозитория и настроим utf-8 в выводе сообщений git (рис. [[1](#fig:001)])

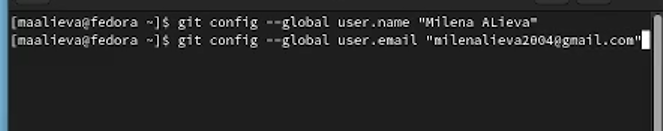


Figure 1: Указание имени и владельца репозитория

1. Зададим имя начальной ветки (будем называть её master), также зададим параметры autocrlf и safecrlf (рис. [[2](#fig:002)])

Figure 2: Установка имени начальной ветки, параметров autocrlf и safecrlf

Figure 2: Установка имени начальной ветки, параметров autocrlf и safecrlf

1. Для последующей идентификации пользователя на сервере репозиториев сгенерируем пару ключей (приватный и открытый) (рис. [[3](#fig:003)])

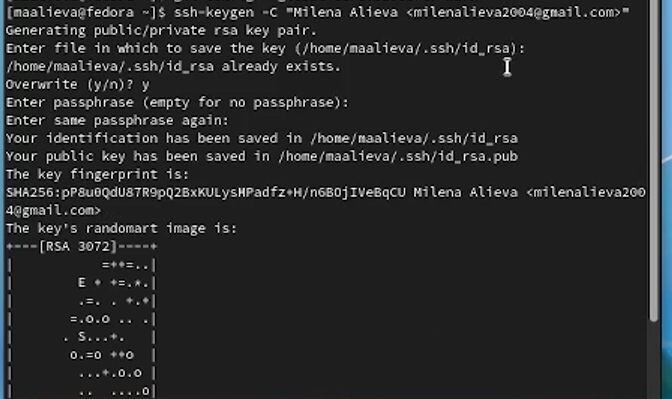


Figure 3: Генерация приватного и открытого ключей SSH

1. Далее загрузим сгенерённый открытый ключ. Для этого зайдём на сайт под своей учетной записью и перейдем в Setting, затем в боковом меню выберем SSH and GPG keys и нажмём на New SSH key. Скопировав из локальной консоли ключ в буфер обмена вставляем ключ в появившееся на сайте поле и указываем для ключа имя (Title)(рис. [[4](#fig:004)])

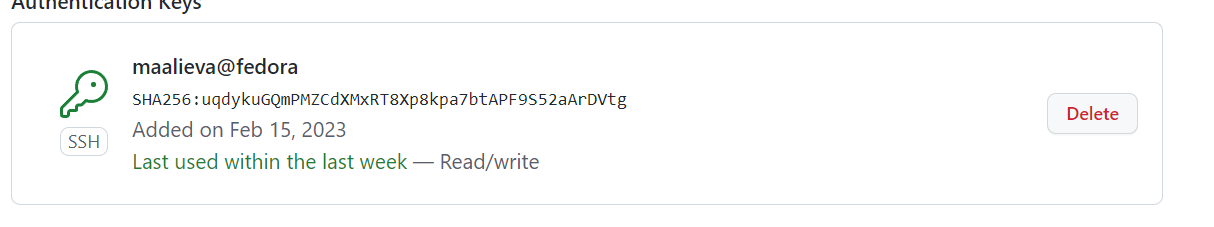


Figure 4: Созданный ключ

1. Для последующей идентификации пользователя на сервере репозиториев сгенерируем Gpg ключ (рис. [[5](#fig:005)])

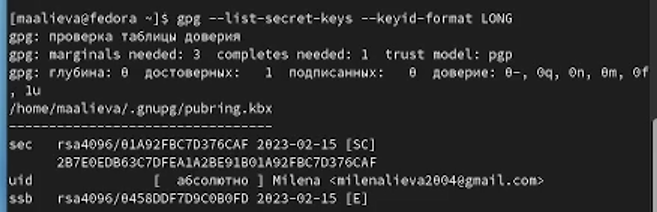


Figure 5: Генерация GPG ключа

1. Сгенерённый GPG ключ (рис. [[6](#fig:006)])

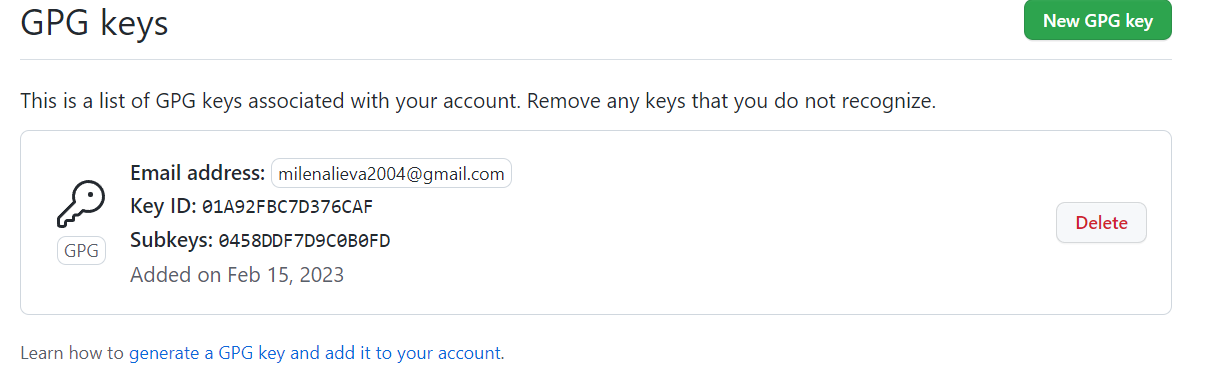


Figure 6: Сгенерённый GPG ключ

1. Авторизация на гитхаб (рис. [[7](#fig:007)])

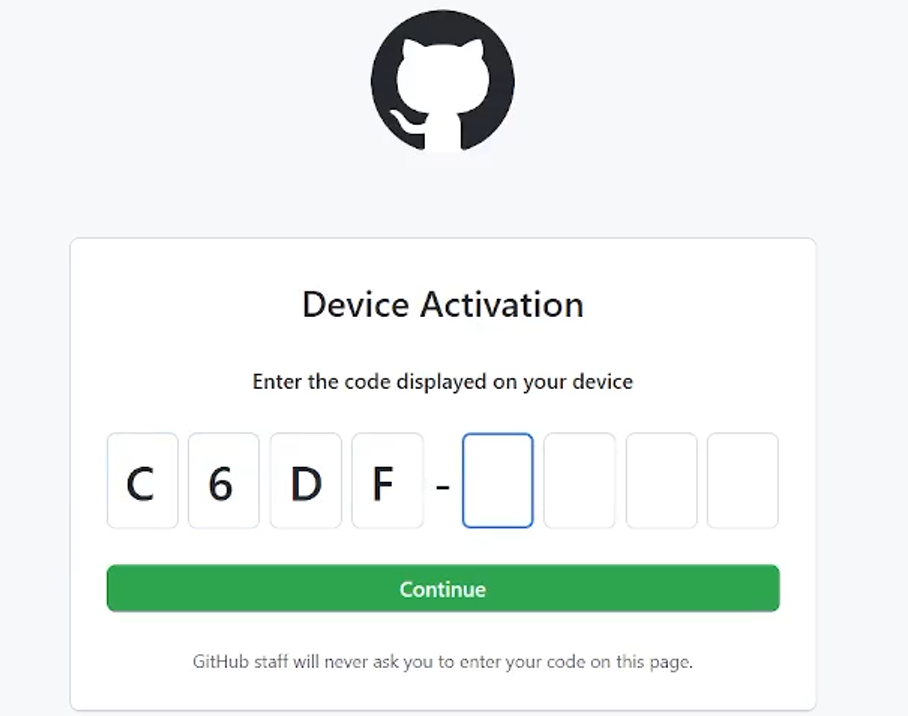


Figure 7: Авторизация

1. Создадим репозиторий на основе шаблона можно через web-интерфейс github. Перейдём на станицу репозитория с шаблоном курса, выберем Use this template. В открывшемся окне зададим имя репозитория (Repository name) study\_2022–2023\_os-intro и создадим репозиторий (Create repository from template) (рис. [[8](#fig:008)])

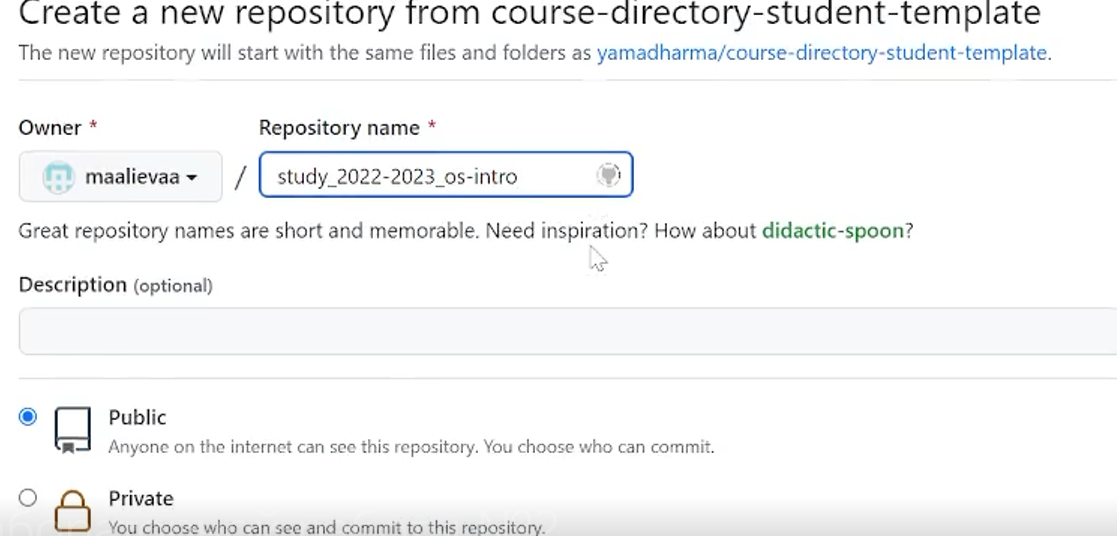


Figure 8: Создание репозитория на основе шаблона

1. Откроем терминал и перейдем в каталог курса. Клонируем созданный репозиторий(рис. [[9](#fig:009)])

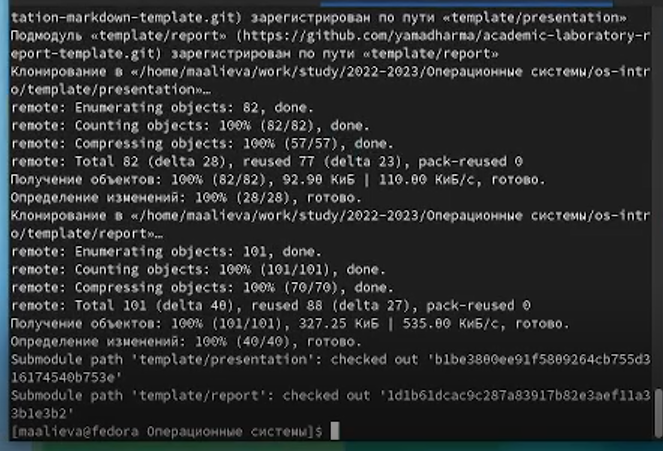


Figure 9: Клонирование репозитория

1. Проверим правильность выполнения, найдем папку “Операционные системы” и пройдем по ней (рис. [[10](#fig:010)])

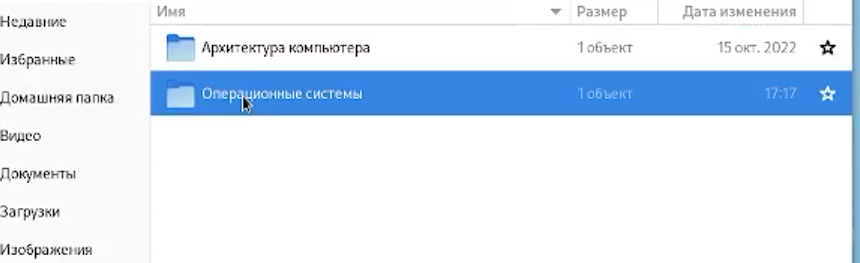


Figure 10: Папка “Операционные системы”

1. Перейдем в каталог курса и удалим лишние файлы. Создадим необходимые каталоги и отправим файлы на сервер (рис. [[11](#fig:011)])

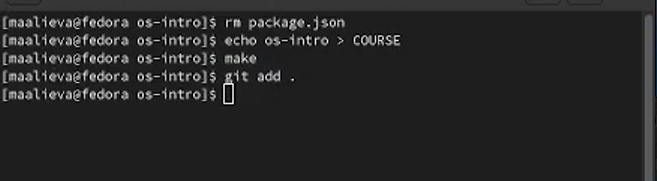


Figure 11: Переход в каталог курса, удаление лишних файлов, создание необходимых каталогов и отправка файлов на сервер

# 4 Ответы на контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются? Система контроля версий — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое. Системы контроля версий (Version Control System, VCS) успешно 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. Репозиторий - хранилище версий - в нем хранятся все документы вместе с историей их изменения и другой служебной информацией. Commit — отслеживание изменений, сохраняет разницу в изменениях. Рабочая копия - копия проекта, связанная с репозиторием (текущее состояние файлов проекта, основанное на версии из хранилища
3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида. В централизованном - одно основное хранилище всего проекта; каждый пользователь копирует необходимые ему файлы из репозитория, изменяет и, затем, добавляет свои изменения обратно В децентрализованном - у каждого пользователя свой вариант (возможно не один) репозитория; присутствует возможность добавлять и забирать изменения из любого репозитория.
4. Опишите действия с VCS при единоличной работе с хранилищем. Необходимо создать и подключить удаленный репозиторий. Затем по мере изменения проекта отправлять эти изменения на сервер.
5. Опишите порядок работы с общим хранилищем VCS. Пользователь получает нужную версию файлов. После того, как он внес необходимые изменения, пользователь размещает новую версию в хранилище.
6. Каковы основные задачи, решаемые инструментальным средством git? Обеспечить удобство командной работы, хранить информацию о всех изменениях.
7. Назовите и дайте краткую характеристику командам git. 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 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
8. Приведите примеры использования при работе с локальным и удалённым репозиториями. Отправка всех произведённых изменений локального дерева в центральный репозиторий: git push
9. Что такое ветви и зачем могут быть нужны ветви (branches)? Ветвь - один из параллельных участков истории в одном хранилище, исходящих их одной версии. Между ветками возможно слияние, что используется для создания новых функций.
10. Как и зачем можно игнорировать некоторые файлы при commit? Есть такие файлы, зачастую временные, которые не нужно добавлять в репозиторий (например, объектные файлы). Так, можно прописать шаблоны игнорируемых при добавлении в репозиторий файлов.

# 5 Выводы

В ходе выполнения лабораторной работы я изучила идеологию и применение средств контроля версий git, а также приобрела практические навыки по работе с системой git.