Лабораторная работа №2

Операционные системы

Александрова Ульяна Вадимовна

Содержание

# 1 Цель работы

Целью работы является изучение идеологии и применение средств контроля версий, а также освоение 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 Теоретическое введение

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.

Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зависимости от настроек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом, привилегированный доступ только одному пользователю, работающему с файлом.

В табл. tbl. 1 приведены основные команды git.

Таблица 1: основные команды git

| Команда | Описание |
| --- | --- |
| git init | Создание основного дерева репозитория |
| git pull | Получение обновлений (изменений) текущего дерева из центрального репозитория |
| git push | Отправка всех произведённых изменений локального дерева в центральный репозиторий |
| git status | Просмотр списка изменённых файлов в текущей директории |
| git diff | Просмотр текущих изменений |
| git add . | Добавить все изменённые и/или созданные файлы и/или каталоги |
| git commit | Сохранить добавленные изменения с внесением комментария через встроенный редактор |
| git checkout | Создание новой ветки, базирующейся на текущей |

# 4 Выполнение лабораторной работы

Я установила git (рис. fig. 1).

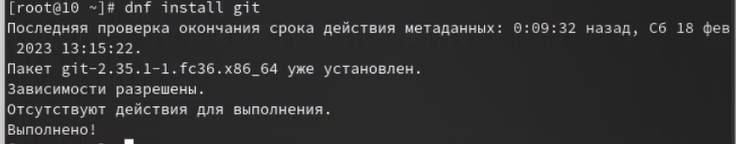


Рис. 1: Установка git

Далее установила gh (рис. fig. 2).

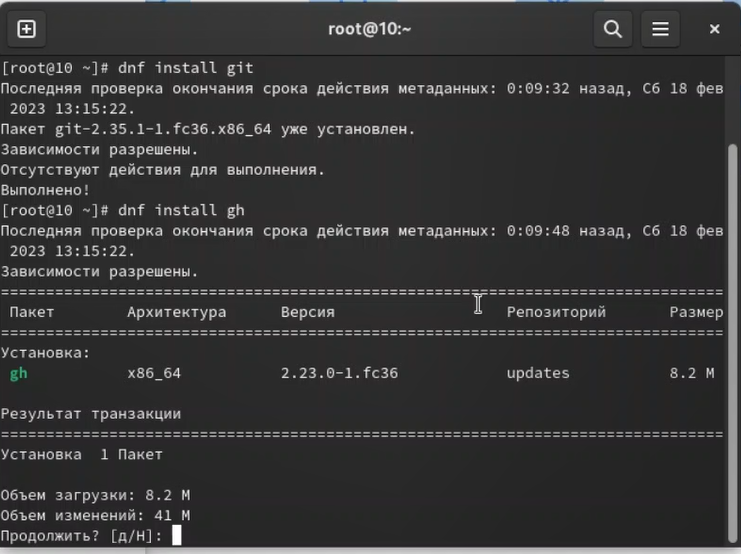


Рис. 2: Установка gh

Так как я уже осуществила базовую настройку git в прошлом семестре, то я сразу перехожу к созданию ключа ssh (рис. fig. 3).

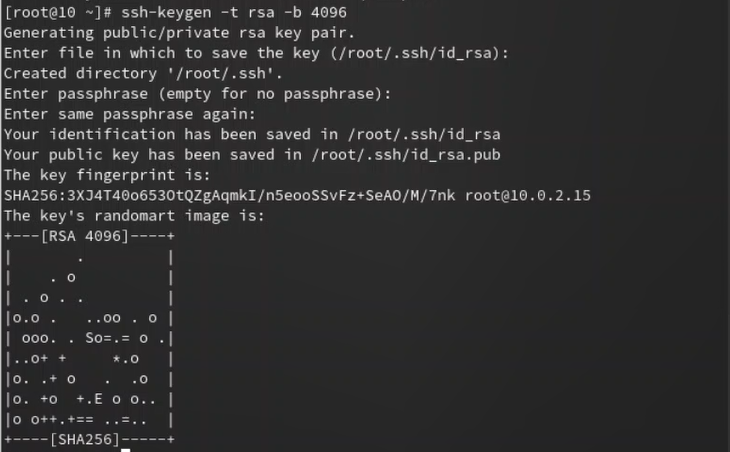


Рис. 3: Генерация ключа ssh

И импортирую его в свои настройки github (рис. fig. 4).

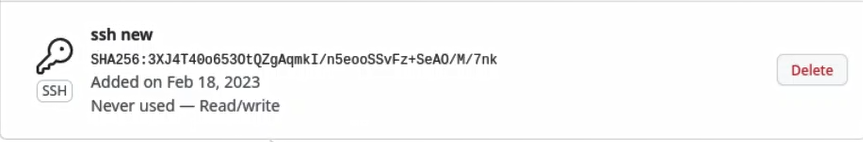


Рис. 4: Импорт ключа

Затем я создаю gpg ключ (рис. fig. 5).

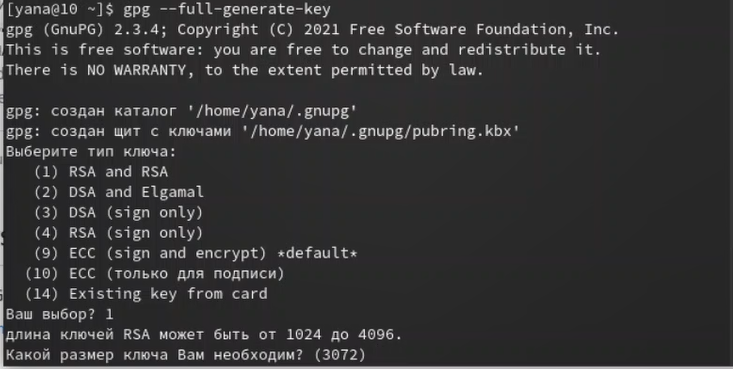


Рис. 5: Генерация gpg ключа

Копирую ключ через специальные утилиты (рис. fig. 6), (рис. fig. 7).

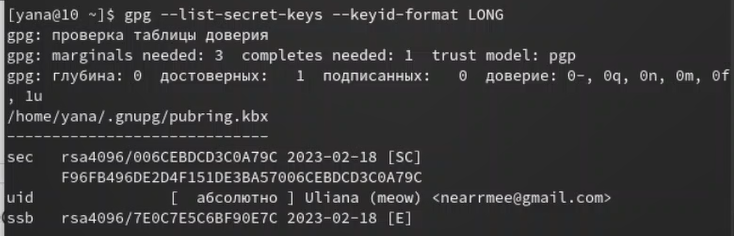


Рис. 6: Вывод списка ключей

Рис. 7: Копирование отпечатка ключа

Рис. 7: Копирование отпечатка ключа

И экспортирую на сайт (рис. fig. 8).

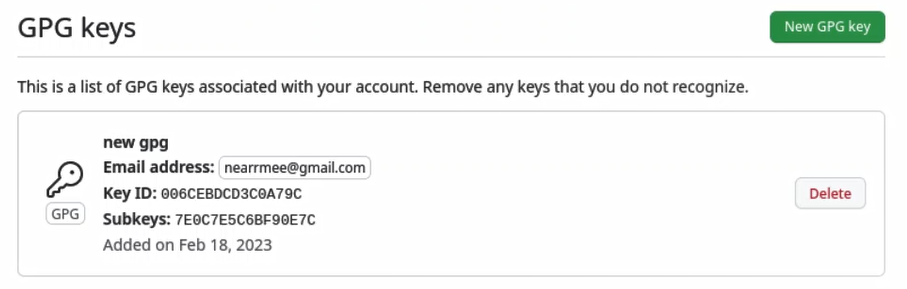


Рис. 8: Импорт

Я настроила автоматические подписи коммитов git (рис. fig. 9).

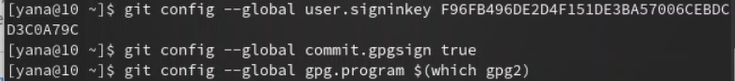


Рис. 9: Настройка commit

Так как я уже работала с сервисом github, мне остается только создать репозиторий курса на основе шаблона (рис. fig. 10), (рис. fig. 11).

Рис. 10: Создание репозитория

Рис. 10: Создание репозитория

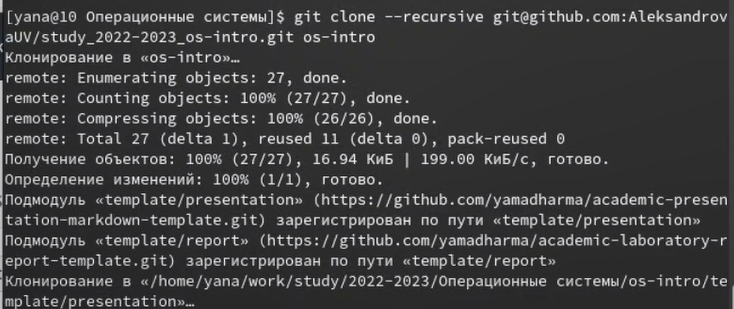


Рис. 11: Копирование репозитория

Далее я настраиваю каталог курса и отправлю файлы на сервер при помощи утилит *git add .*, *git commit -am ’feat(main): make course structure*, *git push* (рис. fig. 12).

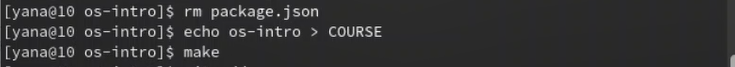


Рис. 12: Название рисунка

# 5 Контрольные вопросы

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

Системы контроля версий (VCS) применяются при работе нескольких человек над одним проектом. Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения.

1. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.

Хранилище - репозиторий, где хранится история изменений, а также док-ты, связанные с работой рабочего пространства.  
Commit - изменения, внесенные пользователями, работающими в одном пространстве.  
История - данные обо всех изменениях, внесенных в проект.  
Рабочая копия - последняя версия проекта.

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

Централизованные VCS - одно основное хранилище всего проекта. Каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет, затем добавляет изменения обратно в хранилище.  
Децентрализованные VCS - у каждого пользователя свой вариант репозитория, есть возможность добавлять и забирать изменения из любого репозитория. В отличие от классических, в децентралиованных системах контроля версий центральный репозиторий не является обязательным.

1. Опишите действия с VCS при единоличной работе с хранилищем.

Как и при совместной работе, создается репозиторий, куда по мере продвижения проекта отправляются изменения.

1. Опишите порядок работы с общим хранилищем VCS.

Пользователю предоставляется доступ к одной из версий проекта, которую он может редактировать и сохранять изменения, доступные другим участникам проекта.

1. Каковы основные задачи, решаемые инструментальным средством git?

Предоставление удобства работы нескольких человек над одним проектом.

1. Назовите и дайте краткую характеристику командам git.

| Команда | Описание |
| --- | --- |
| git init | Создание основного дерева репозитория |
| git pull | Получение обновлений (изменений) текущего дерева из центрального репозитория |
| git push | Отправка всех произведённых изменений локального дерева в центральный репозиторий |
| git status | Просмотр списка изменённых файлов в текущей директории |
| git diff | Просмотр текущих изменений |
| git add . | Добавить все изменённые и/или созданные файлы и/или каталоги |
| git commit | Сохранить добавленные изменения с внесением комментария через встроенный редактор |
| git checkout | Создание новой ветки, базирующейся на текущей |

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

Ветви нужны для того, чтобы программисты могли вести совместную работу над одним проектом вместе, но при этом не мешали друг другу.

# 6 Выводы

Я изучила идеологию и применение средств контроля версий, а также освоение умения по работе с git.