Отчёт по лабораторной работе 2

Архитектура компьютеров

Махкамов Рауфджон НММбд-04-24

Содержание

# 1 Цель работы

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

# 2 Теоретическое введение

## 2.1 Системы контроля версий. Общие понятия

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.

В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять не полную версию изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных.

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.

# 3 Выполнение лабораторной работы

## 3.1 Настройка github

Сначала настраиваю Git на своем компьютере. Создаю пользователя в системе Git и указываю параметры — имя и email, чтобы мои действия были правильно подписаны. (рис. 1)

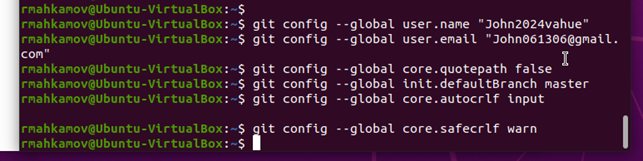


Рис. 1: настраиваю Git

Генерирую SSH-ключи, которые нужны для идентификации на GitHub. (рис. 2) Сохраняю их на компьютере и добавляю публичный ключ в свой профиль на GitHub для установления связи. (рис. 3)

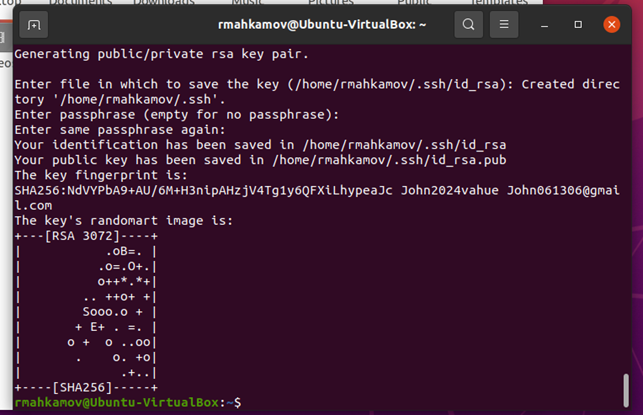


Рис. 2: Генерирую SSH-ключи

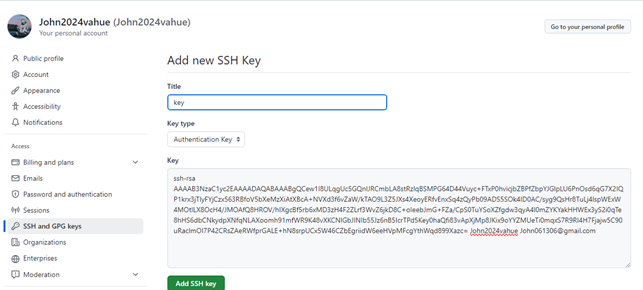


Рис. 3: Добавляю ключ на GitHub

## 3.2 Сознание репозитория курса на основе шаблона

Нахожу нужный репозиторий-шаблон (рис. 4) и делаю из него копию для работы (рис. 5).

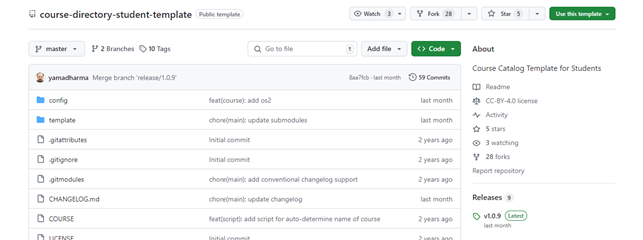


Рис. 4: Репозиторий-шаблон

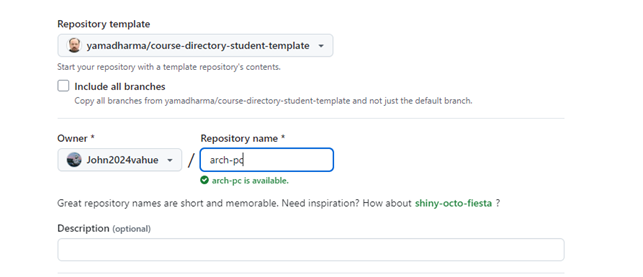


Рис. 5: Копирование шаблона

## 3.3 Настройка каталога курса

Создаю рабочую директорию на своем компьютере для хранения файлов проекта. Клонирую репозиторий с GitHub в эту директорию, чтобы работать с файлами локально. (рис. 6) (рис. 7)

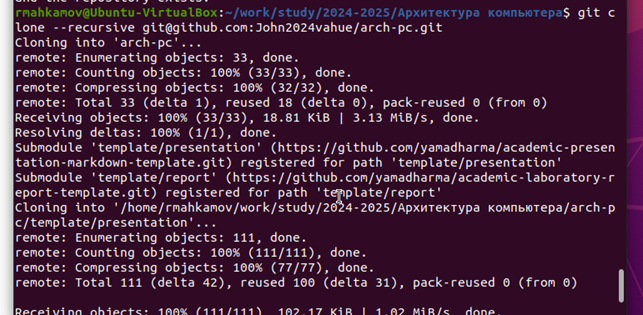


Рис. 6: Клонирование репозитория

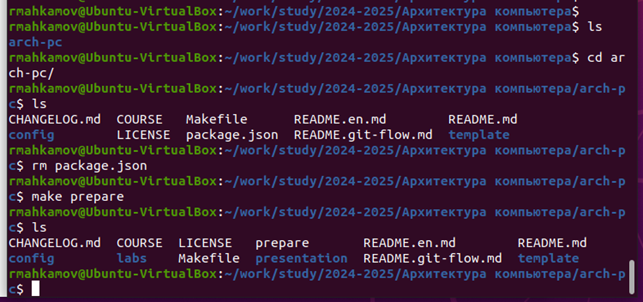


Рис. 7: Подготовка папок

После подготовки структуры добавляю изменения в локальный репозиторий и отправляю их на GitHub командой push. (рис. 8)

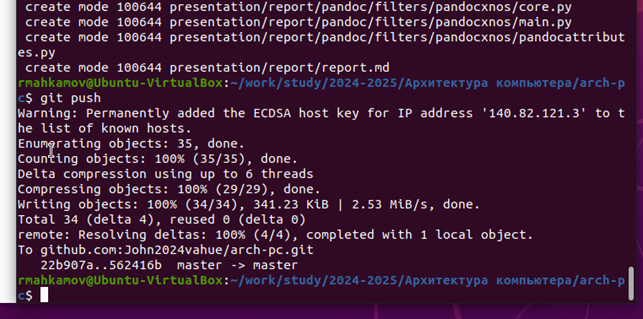


Рис. 8: Отправка изменений в гитхаб

## 3.4 Задание для самостоятельной работы

Загружаю отчёты по выполненным работам в соответствующие папки на GitHub, обновляя репозиторий по мере необходимости. (рис. 9) (рис. 10)

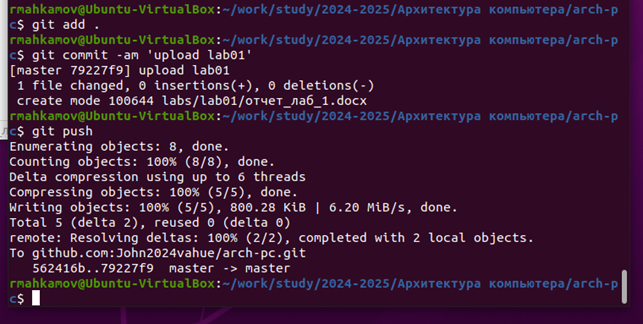


Рис. 9: Отправка изменений в гитхаб

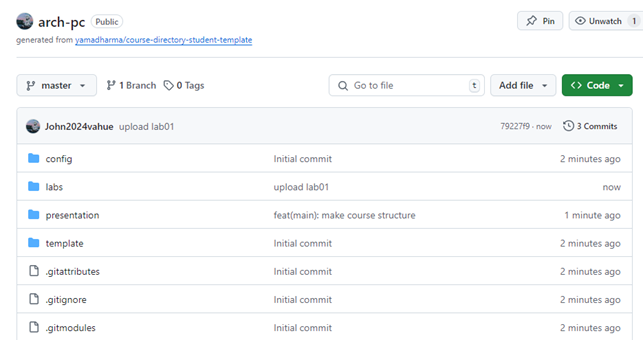


Рис. 10: Обзор репозитория

# 4 Выводы

В ходе выполнения работы изучили работу с GitHub.

# Список литературы

1. Архитектура ЭВМ - Материалы курса
2. Markdown Документация
3. GitHub Документация