Отчёт по лабораторной работе №2

Дисциплина: комьютерные науки и технологии программирования

Логинов Георгий Евгеньевич

Содержание

# 1 Цель работы

Целью данной работы является изучить идеологию и применение средств контроля версий, а также приобрести практические навыки по работе с системой git.

# 2 Задание

1. Настройка GitHub.
2. Базовая настройка Git.
3. Создание SSH-ключа.
4. Создание рабочего пространства и репозитория курса на основе шаблона.
5. Создание репозитория курса на основе шаблона.
6. Настройка каталога курса.
7. Выполнение заданий для самостоятельной работы.

# 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. 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 изменения, сделанные разными участниками, 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. 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. Обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным. Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд. Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями. Благодаря тому, что Git является распределённой системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией.

# 4 Выполнение лабораторной работы

# 5 Выполнение лабораторной работы

## 5.1 Настройка GitHub

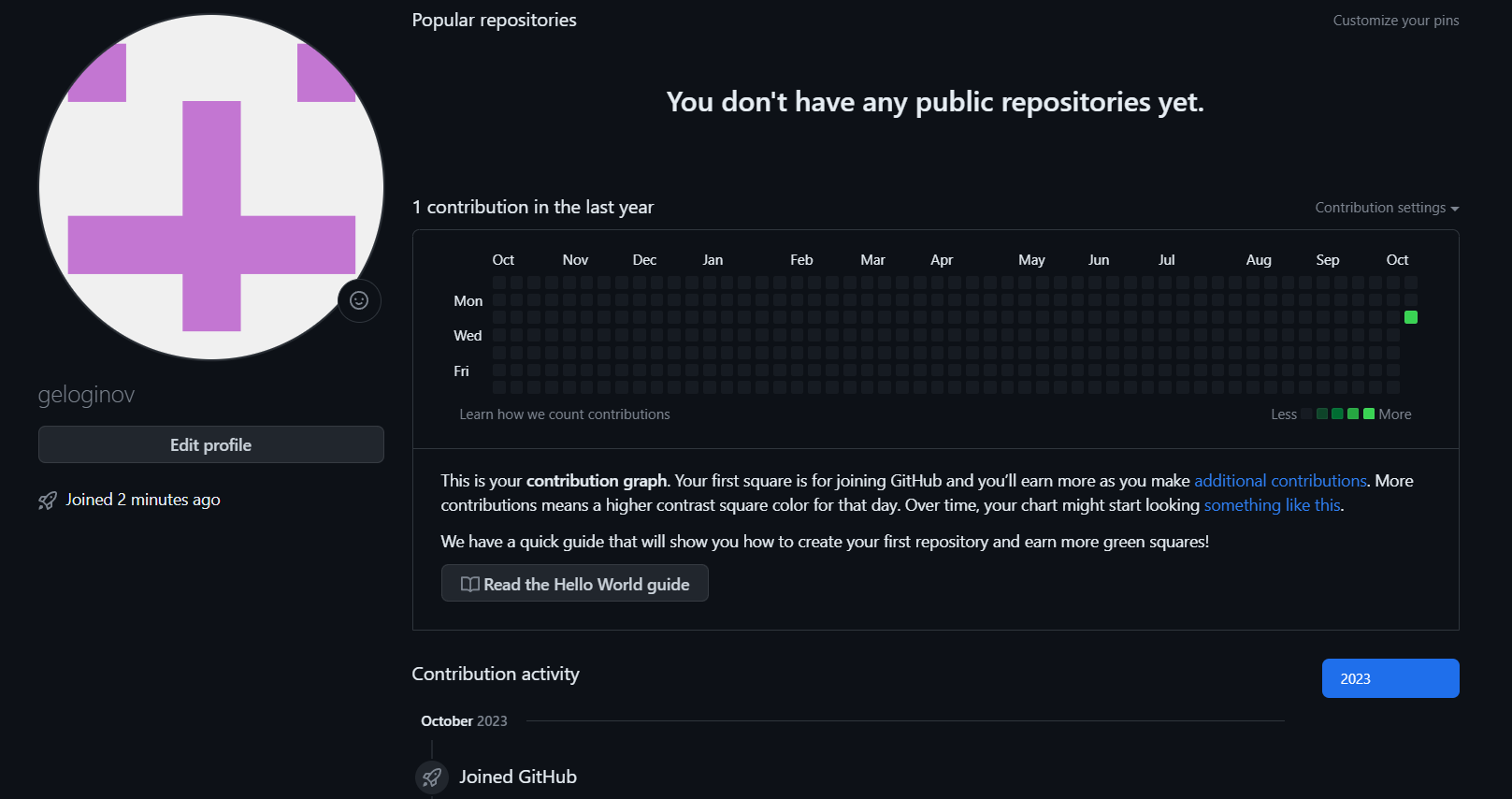


Рисунок 1. Создал учетную запись git

## 5.2 Базовая настройка Git

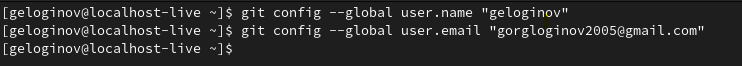


Рисунок 2. Указал имя и email-адрес аккаунта с репозиторием

Рисунок 3. Настроил utf-8 в выводе сообщений git

Рисунок 3. Настроил utf-8 в выводе сообщений git

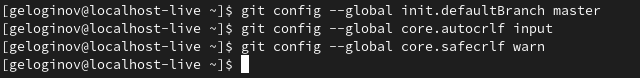


Рисунок 4. Задал имя мастер для начальной ветки и настроила параметры safecrlf и autocrlf

## 5.3 Создание SSH-ключа

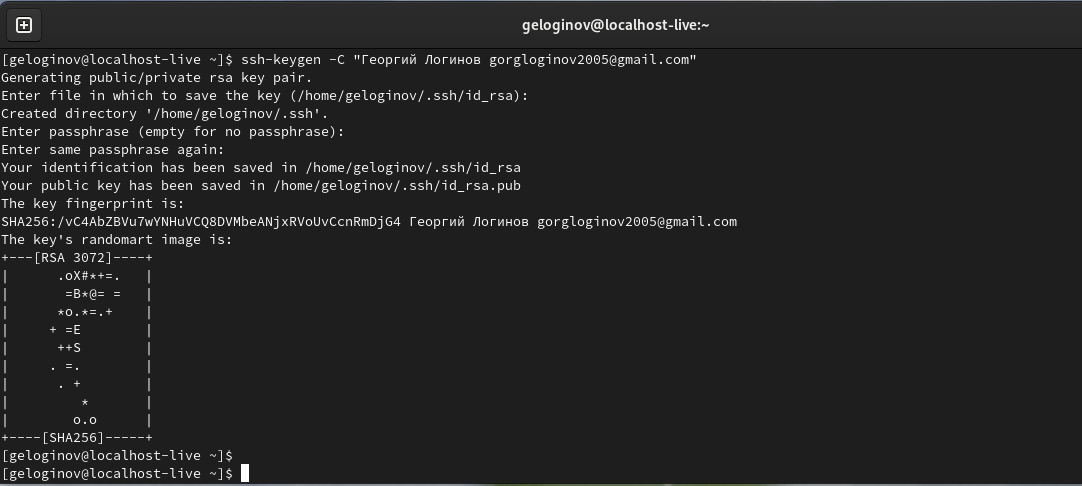


Рисунок 5.Сгенерировал открытый и приватный ключи

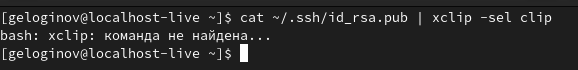


Рисунок 6. Скопировал ключ из консоли в буфер обмена

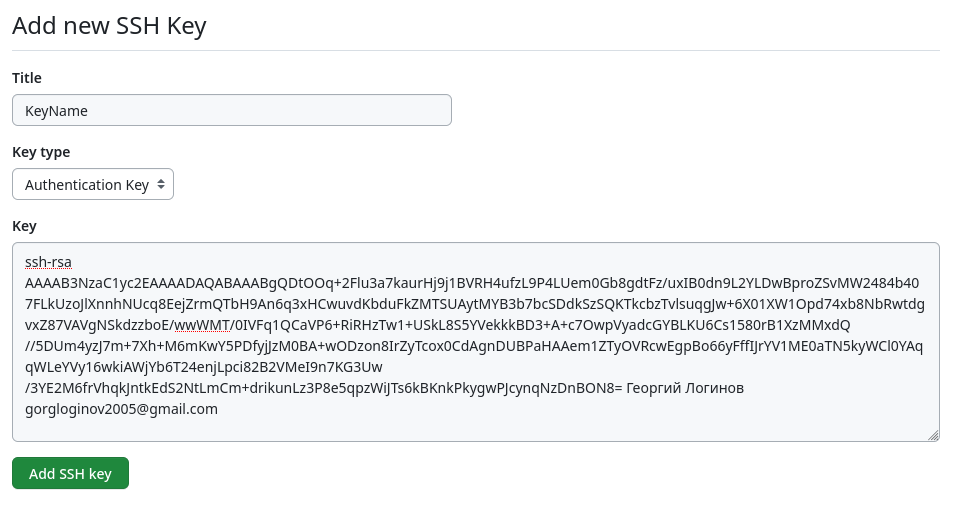


Рисунок 7. Вставил новый ключ

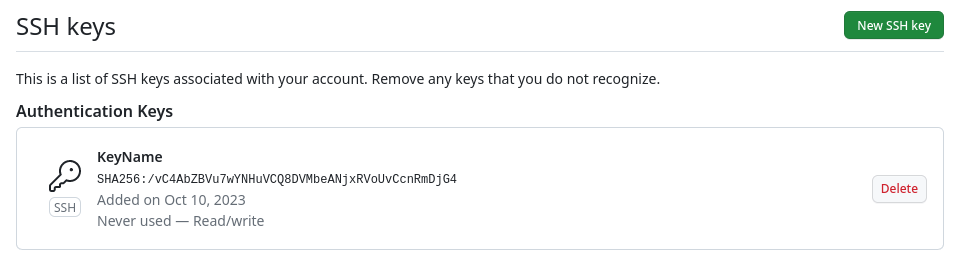


Рисунок 8. Создал ключ

## 5.4 Создание рабочего пространства и репозитория курса на основе шаблона

Рисунок 9. Создал каталог для дисциплины

Рисунок 9. Создал каталог для дисциплины

## 5.5 Создание репозитория курса на основе шаблона



Рисунок 10. Создал репозиторий курса

## 5.6 Настройка каталога курса

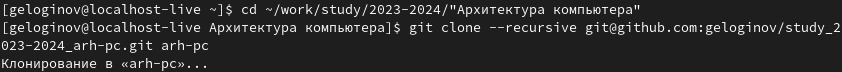


Рисунок 11. Перешел в каталог курса и клонировал репозиторий

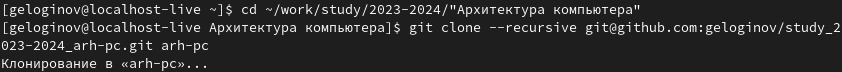


Рисунок 12. Перешел в каталог, удалил лишние файлы и создал каталоги

Рисунок 13. Ввел данные команды

Рисунок 13. Ввел данные команды

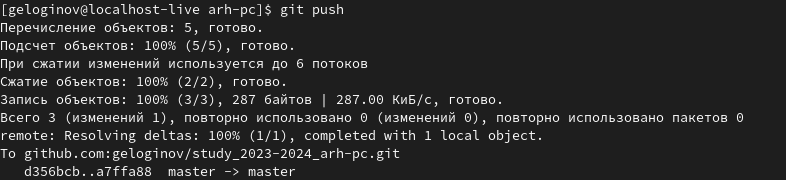


Рисунок 14. Отправил файлы на сервер



Рисунок 15. Проверил корректность создания файлов иерархии рабочего пространства

## 5.7 Задания для самостоятельной работы

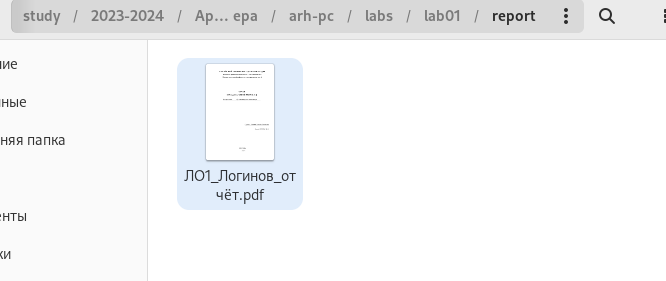


Рисунок 16. Создал отчет по выполнению первой лабораторной работы в соответствующем каталоге



Рисунок 17. Создал отчет по выполнению второй лабораторной работы в соответствующем каталоге

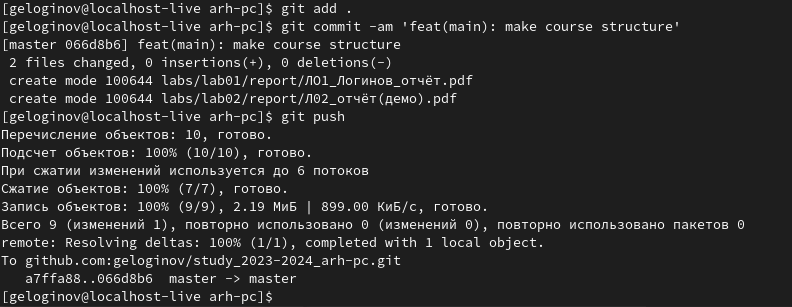


Рисунок 18. Загрузил файлы на github

# 6 Выводы

Здесь кратко описываются итоги проделанной работы.

# Список литературы