Отчёт по лабораторной работе № 2

Королёв Иван Андреевич

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 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 (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.

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. 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.

# 4 Выполнение лабораторной работы

## 4.1 Установка программного обеспечения

* Установка git (рис. [1](#fig:001)).

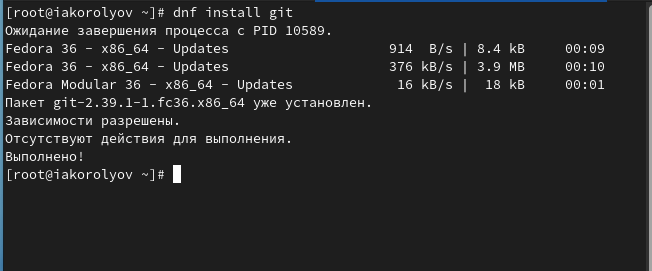


Figure 1: git

* Установка gh (рис. [2](#fig:002)).

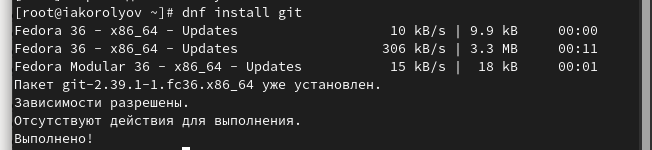


Figure 2: gh

## 4.2 Базовая настройка git

* Зададим имя и email владельца репозитория (рис. [3](#fig:003)).

Figure 3: name and email

Figure 3: name and email

* Настроим utf-8 в выводе сообщений git (рис. [4](#fig:004)).

Figure 4: utf-8

Figure 4: utf-8

* Зададим имя начальной ветки (будем называть её master) (рис. [5](#fig:005)).

Figure 5: master

Figure 5: master

* Параметр autocrlf (рис. [6](#fig:006)).

Figure 6: autocrlf

Figure 6: autocrlf

* Параметр safecrlf (рис. [7](#fig:007)).

Figure 7: safecrlf

Figure 7: safecrlf

## 4.3 Создайте ключи ssh

* Мы уже умеем создавать ssh из прошлого курса Архитектуры компьютеров. (рис. [8](#fig:008)).

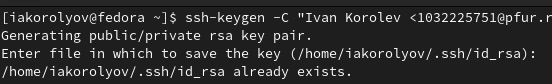


Figure 8: ssh

## 4.4 Создайте ключи pgp

* Генерируем ключ (рис. [9](#fig:009)), (рис. [10](#fig:0010))

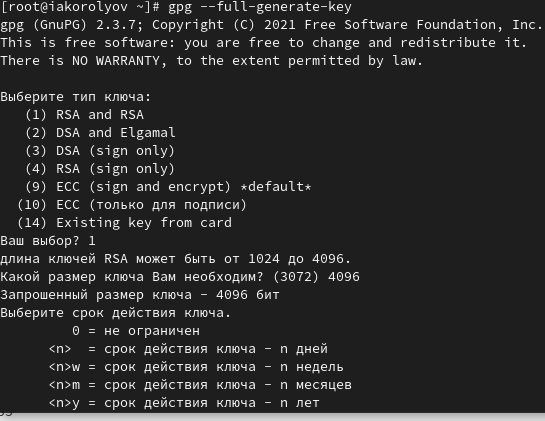


Figure 9: pgp

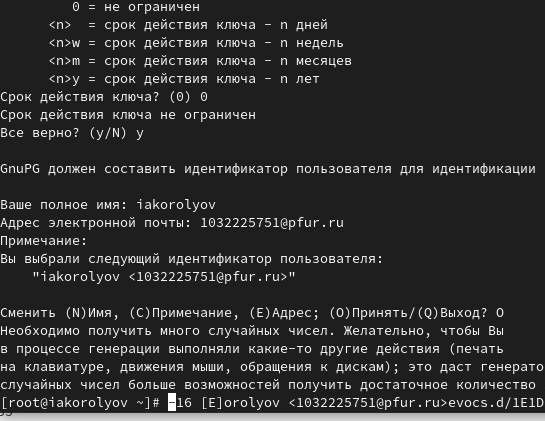


Figure 10: pgp

## 4.5 Настройка github

* У меня создан репозиторий. (рис. [11](#fig:0011))

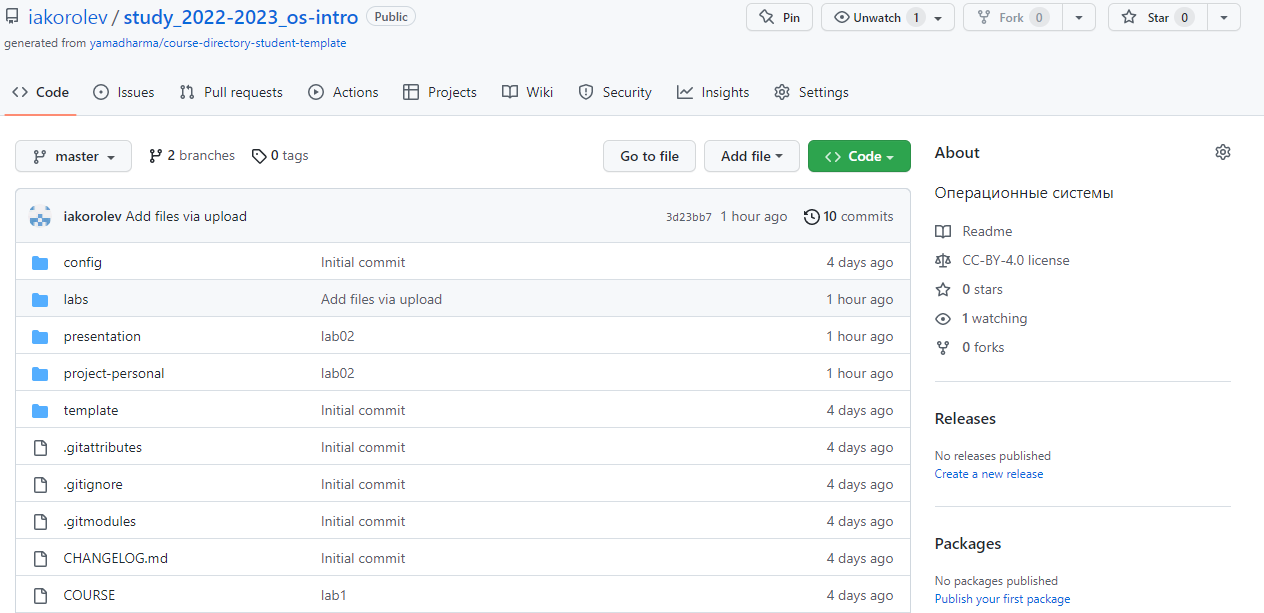


Figure 11: github

## 4.6 Добавление PGP ключа в GitHub

* Выводим список ключей и копируем отпечаток приватного ключа (рис. [12](#fig:0012))

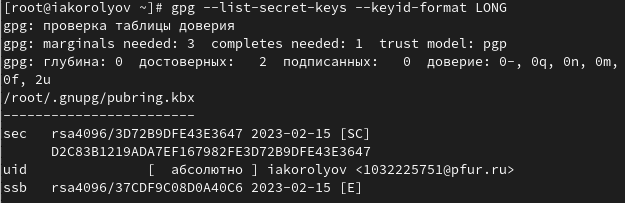


Figure 12: key pgp

* Cкопируйте ваш сгенерированный PGP ключ в буфер обмена (рис. [13](#fig:0013))

Figure 13: copy pgp

Figure 13: copy pgp

* Перейдите в настройки GitHub (https://github.com/settings/keys), нажмите на кнопку New GPG key и вставьте полученный ключ в поле ввода (рис. [14](#fig:0014))

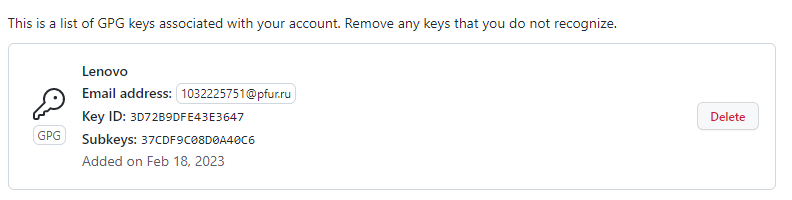


Figure 14: pgp

## 4.7 Настройка автоматических подписей коммитов git

Используя введёный email, укажите Git применять его при подписи коммитов (рис. [15](#fig:0015))

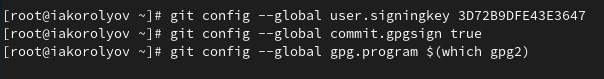


Figure 15: подписи коммитов

## 4.8 Настройка gh

* Авторизоваться в gh (рис. [16](#fig:0016))

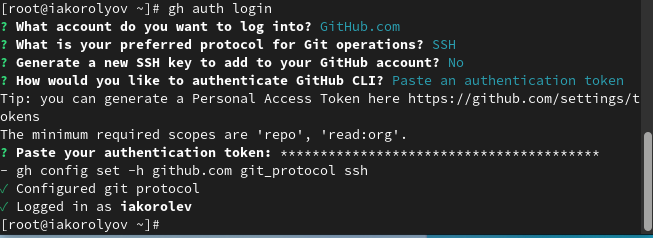


Figure 16: gh

## 4.9 Создание репозитория курса на основе шаблона

* Создать шаблон рабочего пространства (рис. [17](#fig:0017))

Figure 17: gh

Figure 17: gh

* Репозиторий “Операционные системы” создан.

## 4.10 Настройка каталога курса

* Файлы на сервере (рис. [18](#fig:0018)), (рис. [19](#fig:0019))

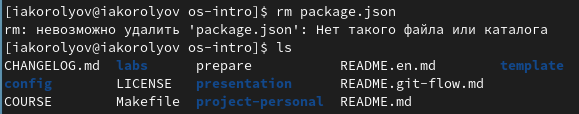


Figure 18: Файлы

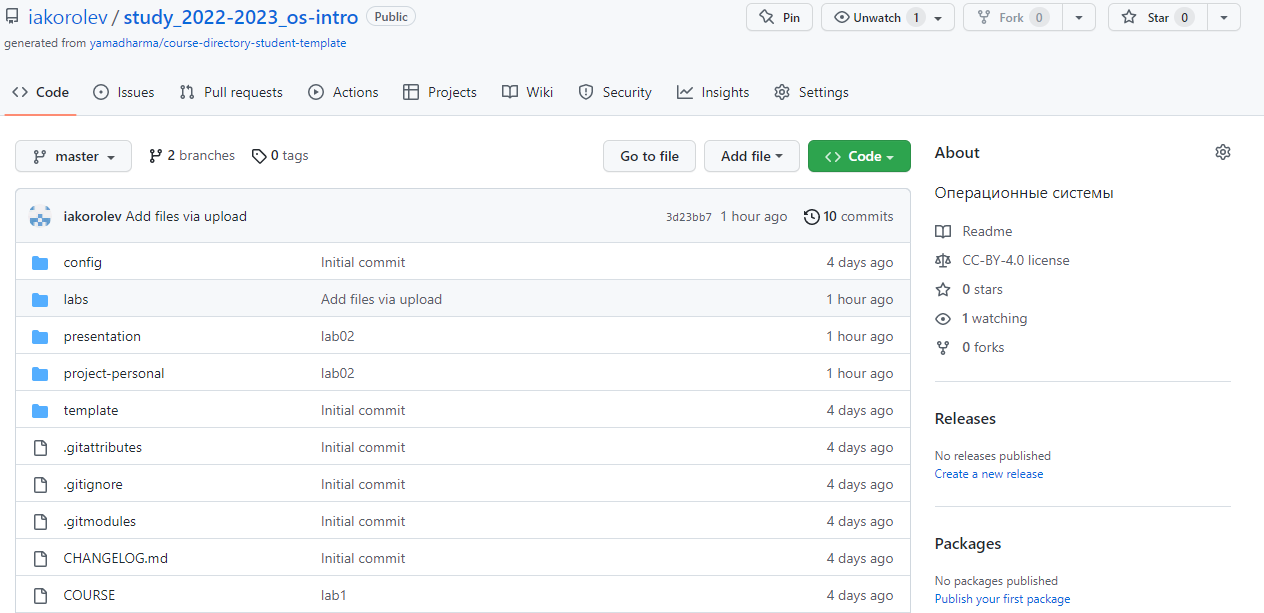


Figure 19: Файлы

# 5 Выводы

Я освоил умения использования git.