Лабораторная работа №2

Операционные системы

Дмитрий Юрьевич Дымченко

Содержание

Список иллюстраций

Список таблиц

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 Выполнение лабораторной работы

Установка git и gh командами dnf install git и dnf install gh (рис. [3.1]).

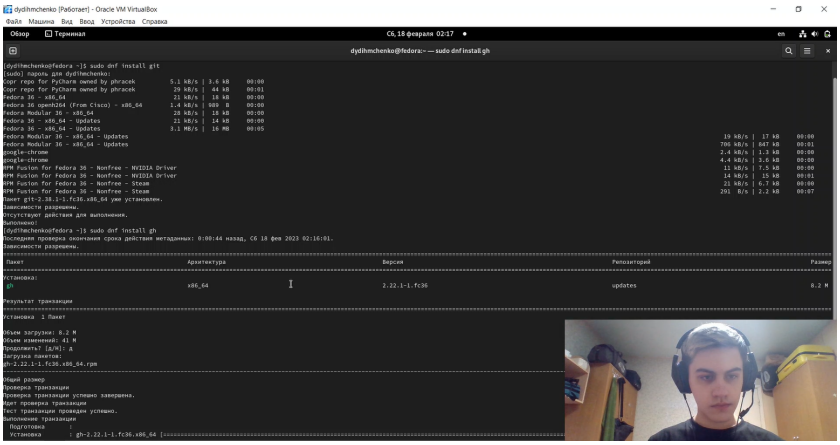

Рис. 3.1: Установка git и gh

Произведем базовую настройку git. (рис. [3.2])

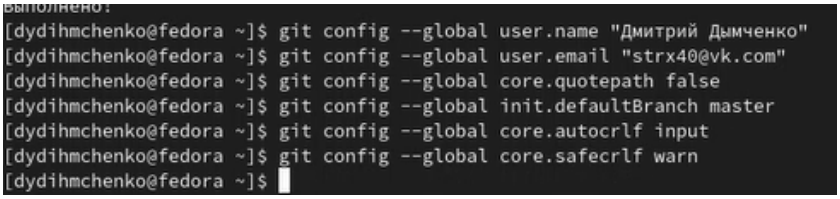

Рис. 3.2: Базовая настройка git

Генерируем ssh-ключи командами ssh-keygen -t rsa -b 4096 и ssh-keygen -t ed25519 (рис. [3.3]), (рис. [3.4])

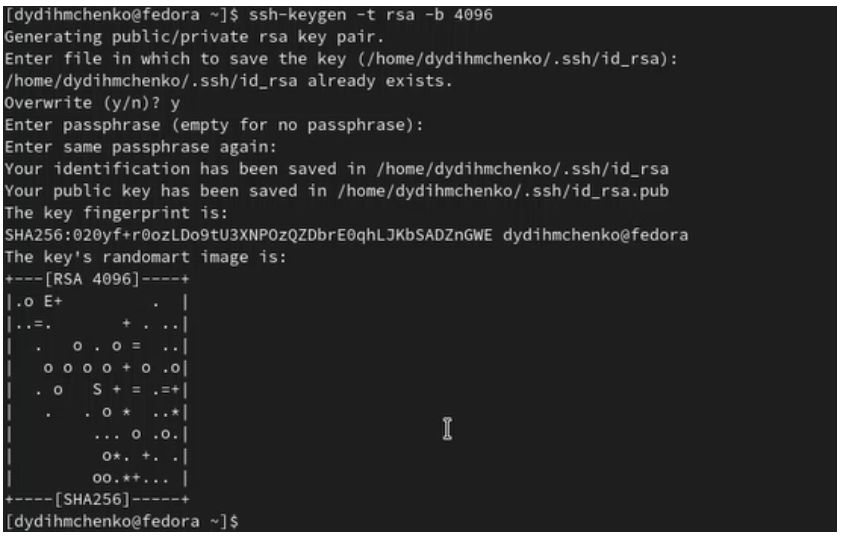

Рис. 3.3: Генерация ssh-ключей

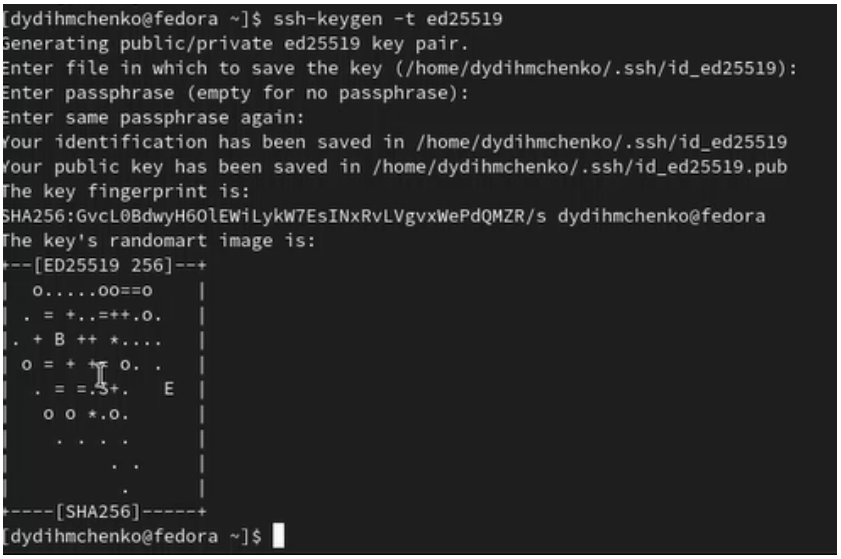

Рис. 3.4: Генерация ssh-ключей

Далее генерируем gpg-ключ командой gpg –full-generate-key (рис. [3.5]).

Рис. 3.5: Генерация gpg-ключа

Привяжем gpg-ключ к учетной записи Github (рис. [3.6]).

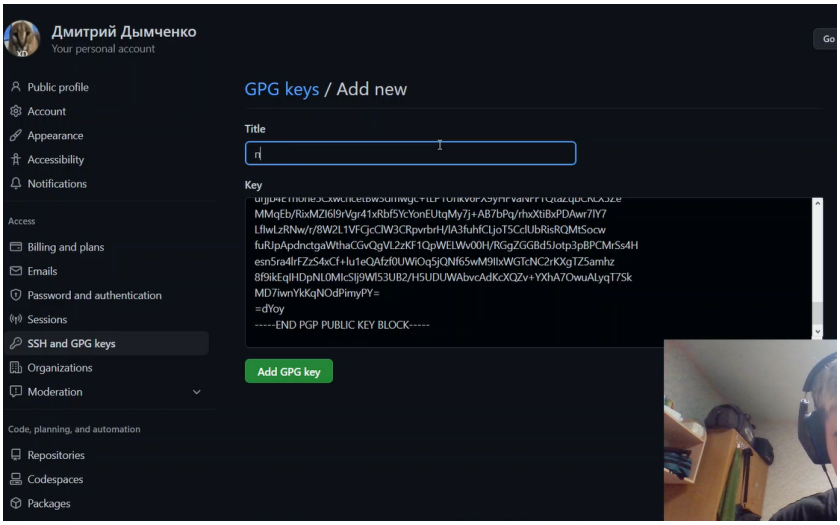

Рис. 3.6: Привязка gpg-ключа

Настраиваем автоматические подписи коммитов git (рис. [3.7]).

```
[dydihmchenko@fedora ~]$ git config --global user.signingkey EA41387AB4B917FB
[dydihmchenko@fedora ~]$ git config --global commit.gpgsign true
[dydihmchenko@fedora ~]$ git config --global gpg.program $(which gpg2)
[dydihmchenko@fedora ~]$
```

Рис. 3.7: Настройка коммитов

Авторизуемся в Github через консоль (рис. [3.8]).

```
[dydihmchenko@fedora ~]$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Paste an authentication token
Tip: you can generate a Personal Access Token here https://github.com/settings/tokens
The minimum required scopes are 'repo', 'read:org', 'workflow'.
? Paste your authentication token: ****************************************
- gh config set -h github.com git_protocol https
✓ Configured git protocol
✓ Logged in as strx40
[dydihmchenko@fedora ~]$
```

Рис. 3.8: Авторизация

Создаем шаблон рабочего пространства (рис. [3.9]).

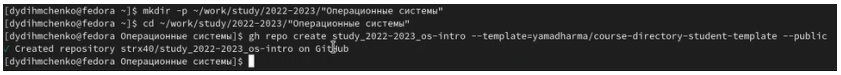

Рис. 3.9: Создание шаблона

Клонируем репозиторий (рис. [3.10]).

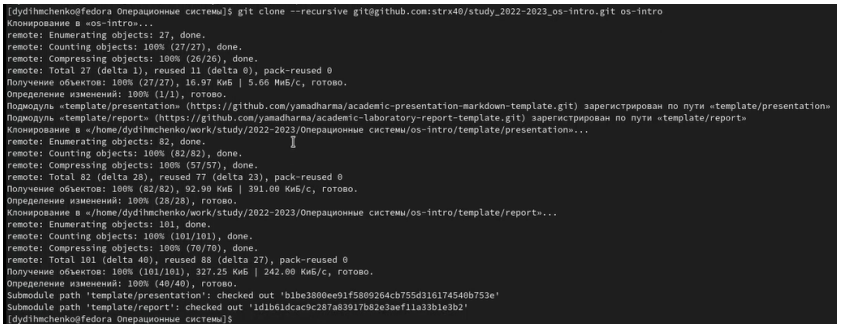

Рис. 3.10: Клонирование репозитория

Проводим настройку каталога курса (рис. [3.11]), (рис. [3.12]).

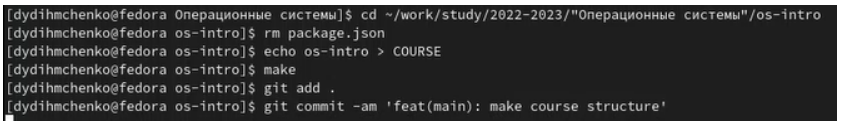

Рис. 3.11: Настройка каталога курса

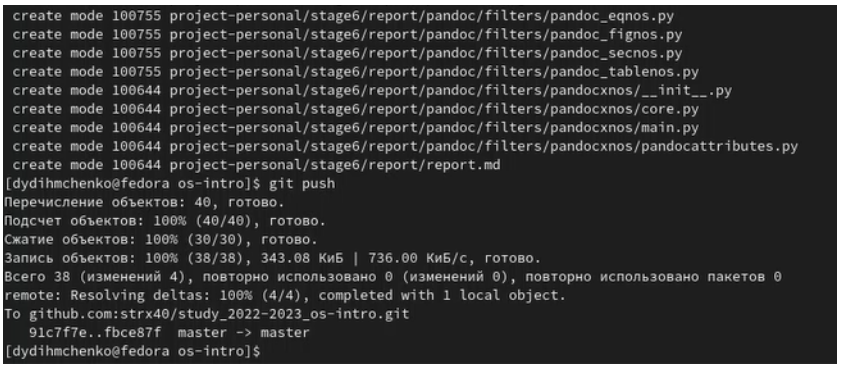

Рис. 3.12: Настройка каталога курса

4 Ответы на контрольные вопросы

1) Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются? Это программное обеспечение для облегчения работы с изменяющейся информацией. VCS 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.

2) Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия. Хранилище (repository), или репозитарий, — место хранения всех версий и служебной информации. Commit («(трудовой) вклад», не переводится) — синоним версии; процесс создания новой версии. История – место, где сохраняются все коммиты, по которым можно посмотреть данные о коммитах. Рабочая копия – текущее состояние файлов проекта, основанное на версии, загруженной из хранилища.

3) Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида. Централизованные VCS: одно основное хранилище всего проекта и каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно. Децентрализованные VCS: у каждого пользователя свой вариант (возможно не один) репозитория.

4) Опишите действия с VCS при единоличной работе с хранилищем.

5) Опишите порядок работы с общим хранилищем VCS.

6) Каковы основные задачи, решаемые инструментальным средством git? Git — это система управления версиями. У Git две основных задачи: первая — хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.

7) Назовите и дайте краткую характеристику командам git. git –version (Проверка версии Git) git init (Инициализировать ваш текущий рабочий каталог как Git-репозиторий) git clone https://www.github.com/username/repo-name (Скопировать существующий удаленный Git-репозиторий) git remote (Просмотреть список текущих удалённых репозиториев Git) git remote -v (Для более подробного вывода) git add my_script.py (Можете указать в команде конкретный файл). git add . (Позволяет охватить все файлы в текущем каталоге, включая файлы, чье имя начинается с точки) git commit -am “Commit message” (Вы можете сжать все индексированные файлы и отправить коммит). git branch (Просмотреть список текущих веток можно с помощью команды branch) git –help (Чтобы узнать больше обо всех доступных параметрах и командах) git push origin master (Передать локальные коммиты в ветку удаленного репозитория).

8) Приведите примеры использования при работе с локальным и удалённым репозиториями.

9) Что такое и зачем могут быть нужны ветви (branches)? Ветки нужны, чтобы несколько программистов могли вести работу над одним и тем же проектом или даже файлом одновременно, при этом не мешая друг другу. Кроме того, ветки используются для тестирования экспериментальных функций: чтобы не повредить основному проекту, создается новая ветка специально для экспериментов.

10) Как и зачем можно игнорировать некоторые файлы при commit? Игнорируемые файлы — это, как правило, артефакты сборки и файлы, генерируемые

машиной из исходных файлов в вашем репозитории, либо файлы, которые по какой-либо иной причине не должны попадать в коммиты.

5 Выводы

В ходе выполнения лабораторной работы я познакомился с принципами работы системы контроля версий, а также освоил некоторые умения работы с git.