Отчёт по лабораторной работе №2

Управление версиями

Хусаинова Динара Айратовна

Содержание

Список иллюстраций

List of Tables

1 Цель работы

Изучить идеологию и применение средств контроля версий и освоить умения по работе с git.

2 Ход работы

1.Создаем учетную запись на GitHub. Устанавливаем git-flow в Fedora Linux (рис. 2.1).

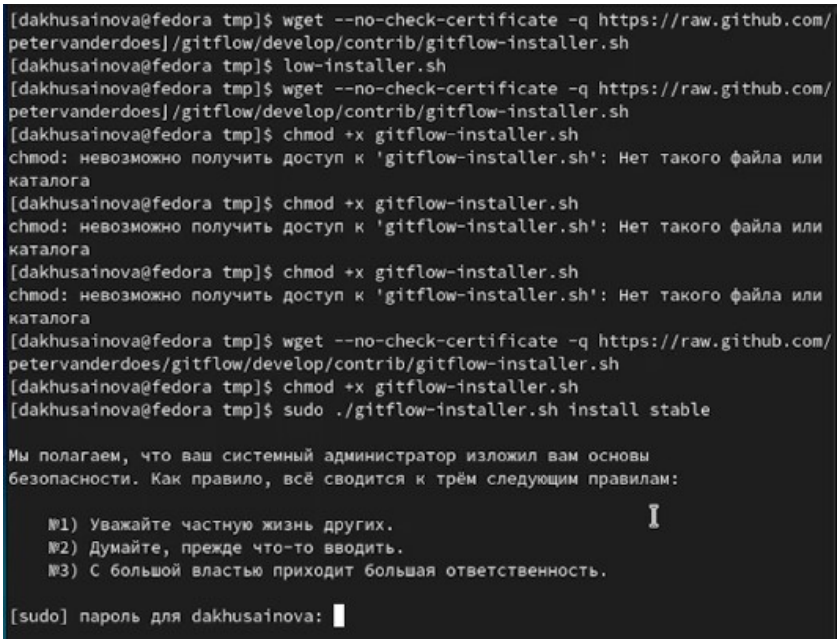

Рис. 2.1: Учетная запись

Устанавливаем gh в Fedora Linux с помощью команды sudo(рис. 2.2).

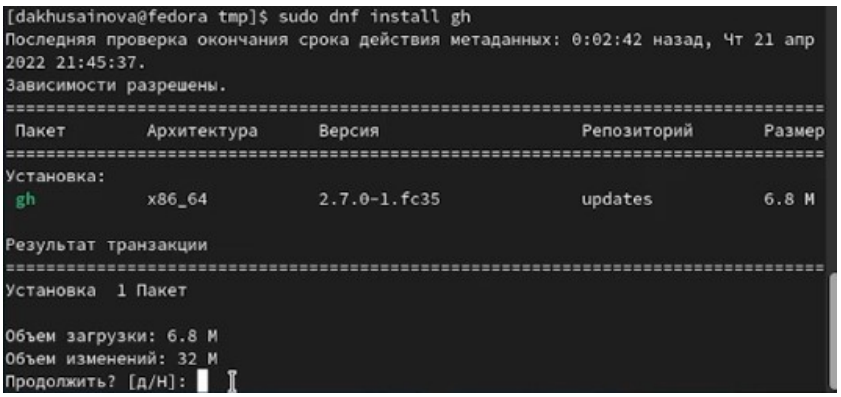

Рис. 2.2: gh

Далее следует базовая настройка git. Используем имя, которое мы вводили при регистрации на GitHub и ту же электронную почту(рис. 2.3).

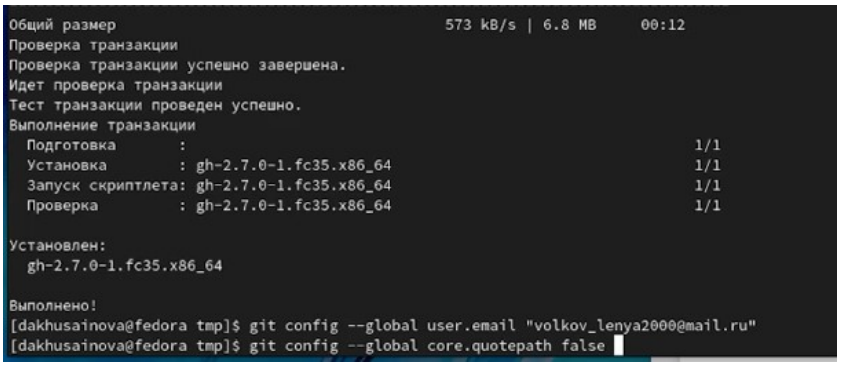

Рис. 2.3: Базовая настройка

Настраиваем utf-8 в выводе сообщений git (рис. 2.4).

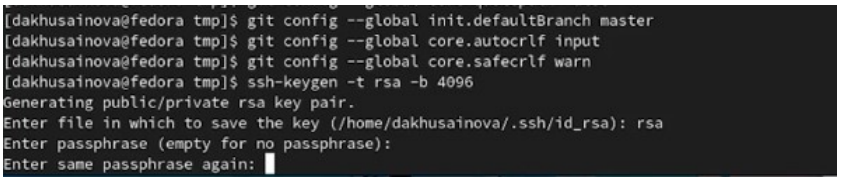

Рис. 2.4: utf-8

При генерации ключа выбираем необходимые опции (рис. 2.5,2.6).

- тип RSA and RSA;
- размер 4096;

- выберите срок действия; значение по умолчанию— 0 (срок действия не истекает никогда).
- GPG запросит личную информацию, которая сохранится в ключе:
- Имя (не менее 5 символов).
- Адрес электронной почты.
- При вводе email убедитесь, что он соответствует адресу, используемому на GitHub.
- Комментарий

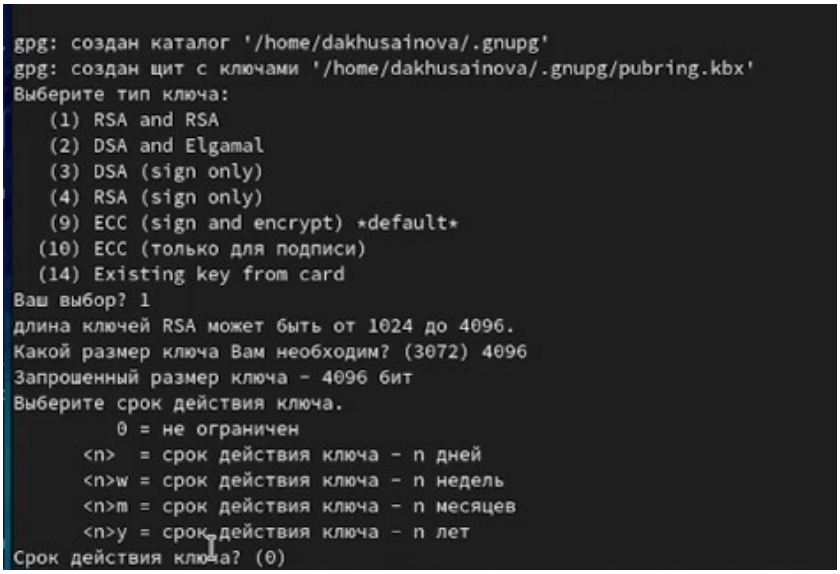

Рис. 2.5: Генерация ключа

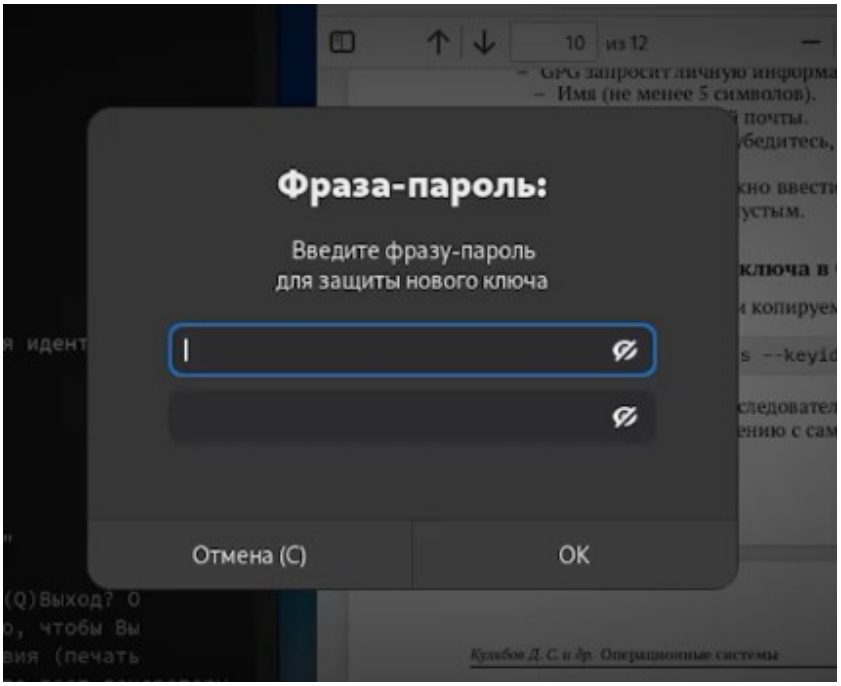

Рис. 2.6: Выбираем опции

Выводим список ключей и копируем отпечаток приватного ключа, устанавливаем необходимый пакет для дальнейших действий с ключом (рис. 2.7 -2.8).

Рис. 2.7: Список ключей

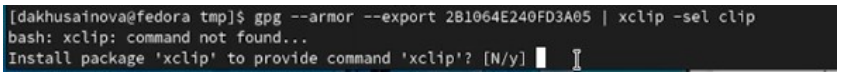

Рис. 2.8: Приватный ключ

Заходим на сам сайт GitHub, копируем сгенерированный PGP ключ в буфер обмена и вставляем полученный ключ в поле ввода (рис.2.9,2.10,2.11).

Рис. 2.9: Сайт

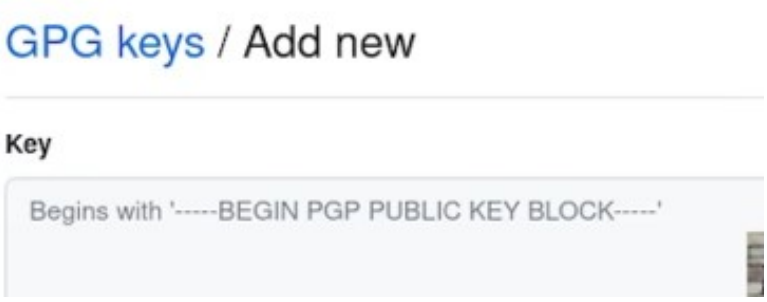

Рис. 2.10: Открываем

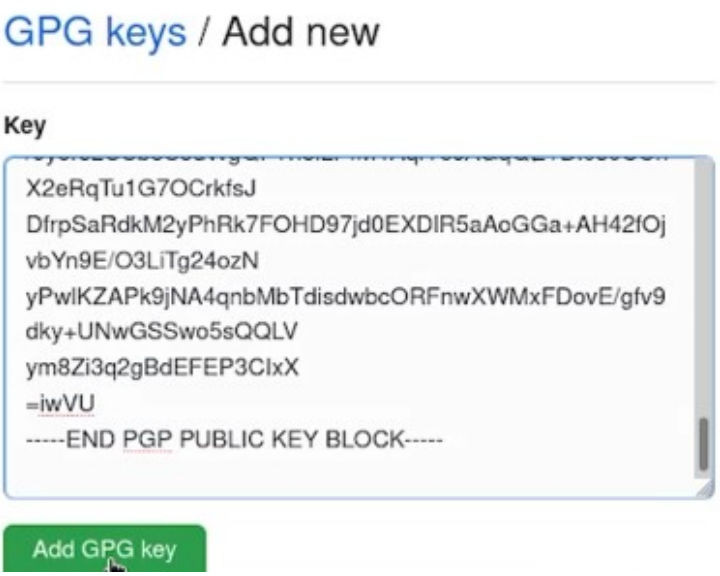

Рис. 2.11: Вставляем полученный ключ

Настраиваем автоматические подписи коммитов git, отвечаем на вопросы, выбирая GitHub,SSH,yes,вводим пароль, «залогиниться через браузер»(рис.2.12-2.13).

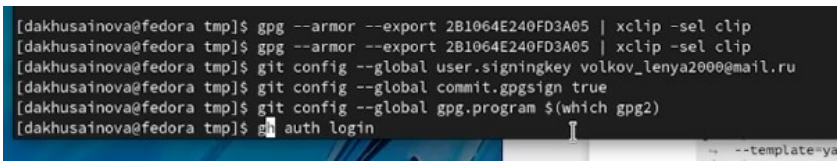

Рис. 2.12: Коммиты git

Рис. 2.13: Отвечаем на необходимые вопросы

Получаем код в терминале и вводим его на сайте(рис.2.14).

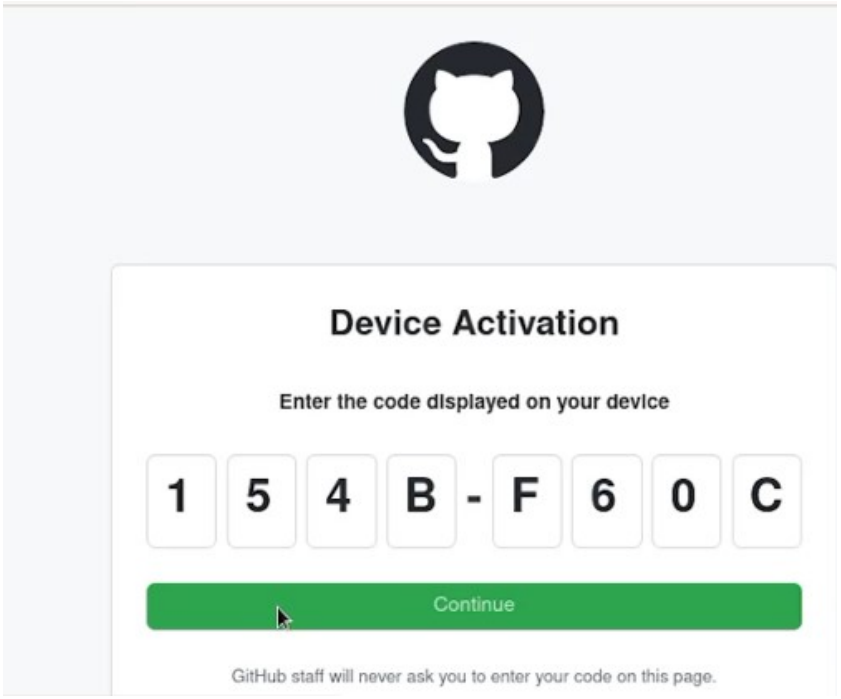

Рис. 2.14: Пишем код из терминала

Переходим в каталог курса, удаляем ненужные файлы, создаем, создаем каталог os-intro и отправляем файлы на сервер(рис. 2.15,2.16).

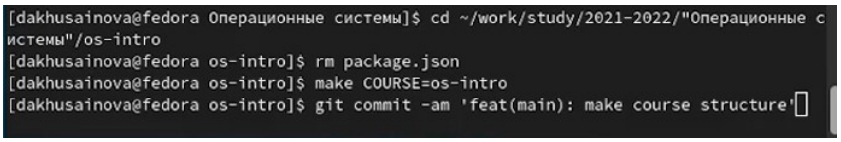

Рис. 2.15: Проделываем выше описанные действия

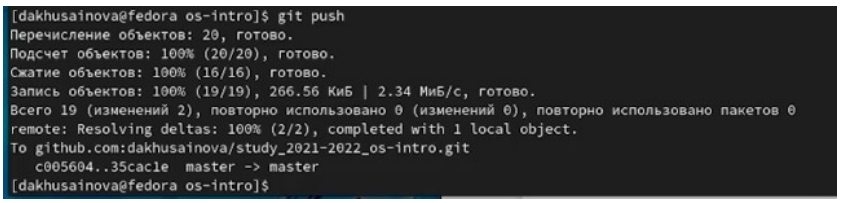

Рис. 2.16: Отправляем файлы

3 Контрольные вопросы

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

Система контроля версий (Version Control System, VCS) — программное обеспечение для облегчения работы с изменяющейся информацией. u 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.

- Commit (Коммит) – сохранение изменений в репозиторий.
- Хранилище (repository), или репозитарий, — место хранения всех версий и служебной информации.
- История — список всех изменений проекта с возможностью отката в любую точку истории.
- Рабочая копия (working copy) — текущее состояние файлов проекта (любой версии), полученных из хранилища и, возможно, измененных.

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

Централизованные системы — это системы, которые используют архитектуру клиент / сервер, где один или несколько клиентских узлов напрямую подключены к центральному серверу. Это наиболее часто используемый тип систем во многих

организациях, где клиент отправляет запрос на сервер компании и получает ответ. - Wikipedia. Рассмотрим огромный сервер, на который мы отправляем наши запросы, и сервер отвечает запрашиваемой статьей. Предположим, мы ввели поисковый запрос «нездоровая пища» в строке поиска Википедии. Этот поисковый запрос отправляется как запрос на серверы Википедии (в основном, расположенные в штате Вирджиния, США), которые затем возвращают статьи, основанные на релевантности. В этой ситуации мы являемся клиентским узлом, серверы Википедии являются центральным сервером.

В децентрализованных системах каждый узел принимает свое собственное решение. Конечное поведение системы является совокупностью решений отдельных узлов. Обратите внимание, что нет единого объекта, который получает и отвечает на запрос - Bitcoin. Давайте возьмем биткойны, например, потому что это самый популярный пример использования децентрализованных систем. Ни одна организация / организация не владеет сетью биткойнов. Сеть представляет собой сумму всех узлов, которые общаются друг с другом для поддержания количества биткойнов, которое есть у каждого владельца счета.

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

Инициализация системы управления версиями git через git init. Работа над проектом используя git-flow для отдельных частей проекта. Git commit для фиксации изменений. При необходимости использование удаленного сервера для хранения с помощью remote и git push. Удаленный сервер также позволяет работать с нескольких устройств с использованием git pull.

5. Опишите порядок работы с общим хранилищем VCS.

При существующей версии проекта в хранилище, скопировать его оттуда через git pull. Использовать git-flow для работы над частями проекта. После окончания работы зафиксировать изменения через git commit и загрузить в хранилище через git push.

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. Ведение истории изменений, фиксирование изменений, совмещение версий, веток и др., а также откат к прошлым версиям.

7. Назовите и дайте краткую характеристику командам git.

- git init — инициализация проекта с системой контроля версий.
- git add — добавление файла/директории в систему контроля версий как отслеживаемое.
- git commit — фиксация изменений в отслеживаемых файлах.

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

При работе с локальным репозиторием все изменения хранятся локально и не выгружаются на удаленный сервер. Не требуется использование команд push, pull, remote и т.д. При работе с удаленным репозиторием для отображения изменения на удаленном репозитории и его актуализации, последние изменения должны быть загружены на удаленный сервер.

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

Ветка (англ. branch) — это последовательность коммитов, в которой ведётся параллельная разработка какого-либо функционала Основная ветка– master Ветки в GIT. Показать все ветки, существующие в репозитарии git branch. Создать ветку git branch имя.

10. Как и зачем можно игнорировать некоторые файлы при commit? Игнорирование файлов при commit происходит с помощью .gitignore файла.

В нем указываются пути, названия, расширения и другие идентификации нежелательных объектов которые не будут учитываться в commit. Это полезно для исключения как “мусорных” файлов, которые не являются значимой частью проекта, а также конфиденциальных файлов, которые содержат в себе приватную информацию, такую как пароли и токены.

4 Вывод

Мы изучили идеологию и применение средств контроля версий и освоили умения по работе с git.