

Отчёт по лабораторной работе № 2

Операционные системы

Паулу Антонью Жоау

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
3.1	Установка программного обеспечения	6
3.2	Базовая настройка git	7
3.3	Создали ключи ssh	7
3.4	Создали ключи pgr	8
3.5	Настройка github	10
3.6	Добавление PGP ключа в GitHub	11
3.7	Настройка автоматических подписей коммитов git	12
3.8	Настройка gh	12
3.9	Создание репозитория курса на основе шаблона	13
3.10	Настройка каталога курса	14
4	Выводы	16
5	Ответы на контрольные вопросы	17

Список иллюстраций

3.1	6
3.2	6
3.3	7
3.4	7
3.5	7
3.6	7
3.7	7
3.8	8
3.9	8
3.10	9
3.11	10
3.12	10
3.13	11
3.14	11
3.15	11
3.16	12
3.17	12
3.18	12
3.19	13
3.20	13
3.21	13
3.22	14
3.23	14
3.24	14
3.25	14
3.26	15
3.27	15
5.1	18

1 Цель работы

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

2 Задание

- Установить и настроить ПО для работы с git.

3 Выполнение лабораторной работы

3.1 Установка программного обеспечения

Установили git:(рис. [3.1])

```
[azpaulu@fedora ~]$ sudo -i
[sudo] пароль для azpaulu:
[root@fedora ~]# dnf install git
Последняя проверка окончания срока действия метаданных: 2:48:15 назад, Пт 23 июн 2023 22:22:48.
Пакет git-2.35.1-1.fc36.x86_64 уже установлен.
Зависимости разрешены.
Отсутствуют действия для выполнения.
Выполнено!
```

Рис. 3.1: .

Установили gh:(рис. [3.2])

```
Выполнено:
[root@fedora ~]# dnf install gh
Последняя проверка окончания срока действия метаданных: 2:48:43 назад, Пт 23 июн 2023 22:22:48.
Зависимости разрешены.
=====
Пакет      Архитектура  Версия      Репозиторий  Размер
=====
Установка:
gh          x86_64       2.23.0-1.fc36  updates      8.2 М
=====
Результат транзакции
=====
Установка 1 Пакет

Объем загрузки: 8.2 М
Объем изменений: 41 М
Продолжить? [д/н]: 1
Продолжить? [д/н]: д
Загрузка пакетов:
gh-2.23.0-1.fc36.x86_64 0% [
```

Рис. 3.2: .

3.2 Базовая настройка git

Задали имя и email владельца репозитория: (рис. [3.3])

```
[azpaulu@fedora ~]$ git config --global user.name "<antoniopaulo11>"
[azpaulu@fedora ~]$ git config --global user.email "<antoniopaulo33@icloud.com>"
```

Рис. 3.3: .

Настроили utf-8 в выводе сообщений git:(рис. [3.4])

```
[azpaulu@fedora ~]$ git config --global core.quotePath false
```

Рис. 3.4: .

Настроили верификацию и подписание коммитов git. Задали имя начальной ветки (будем называть её master).(рис. [3.5])

```
[azpaulu@fedora ~]$ git config --global init.defaultBranch master
```

Рис. 3.5: .

Параметр autocrlf:(рис. [3.6])

```
[azpaulu@fedora ~]$ git config --global core.autocrlf input
```

Рис. 3.6: .

Параметр safecrlf: (рис. [3.7])

```
[azpaulu@fedora ~]$ git config --global core.safecrlf warn
```

Рис. 3.7: .

3.3 Создали ключи ssh

по алгоритму rsa с ключём размером 4096 бит: (рис. [3.8])

```
[azpaulu@fedora ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/azpaulu/.ssh/id_rsa):
/home/azpaulu/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/azpaulu/.ssh/id_rsa
Your public key has been saved in /home/azpaulu/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:KZeTsin6LJ7YAGrYHqS/9DkroYUJDXy0Hl0DrCQedg azpaulu@fedora
The key's randomart image is:
+---[RSA 4096]-----+
|
| =
|B E
|o0 . +
|0.= o S
|*@o * .
|==*.. o
|=Bo*.o
|+o0=*o
+---[SHA256]-----+
```

Рис. 3.8: .

по алгоритму ed25519: (рис. [3.9])

```
+---[SHA256]-----+
[azpaulu@fedora ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/azpaulu/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/azpaulu/.ssh/id_ed25519
Your public key has been saved in /home/azpaulu/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:59pA4qovzSDNxPLZ0GQpinWD63cu9z0FW+EH0MoLYUo azpaulu@fedora
The key's randomart image is:
+---[ED25519 256]---+
| . . .o |
| + * E o + |
|,+ B o o o o |
|+ = . . . + o . |
| B + . S., = . |
| , B o..o oo . |
| . = o. . . . |
| . +.o =. |
| .+oo .o o. |
+---[SHA256]-----+
```

Рис. 3.9: .

3.4 Создали ключи рдр

Сгенерировали ключ (рис. [3.10])

Из предложенных опций выбирали: тип RSA and RSA; размер 4096; выбрали срок действия; значение по умолчанию — 0 (срок действия не истекает никогда).

GPG запросил личную информацию, которая сохранится в ключе: Имя. Адрес электронной почты. При вводе email убедились, что он соответствует адресу, используемому на GitHub. (рис. [3.11])

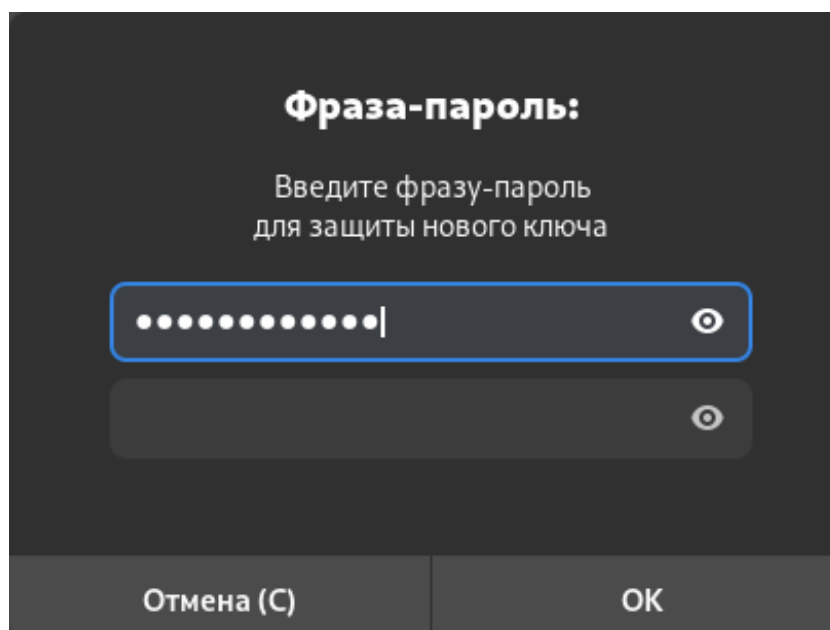


Рис. 3.10: .

```
[azpaulu@fedora ~]$ gpg --full-generate-key
gpg (GnuPG) 2.3.4; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: создан каталог '/home/azpaulu/.gnupg'
gpg: создан шит с ключами '/home/azpaulu/.gnupg/pubring.kbx'
Выберите тип ключа:
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) *default*
 (10) ECC (только для подписи)
 (14) Existing key from card
Ваш выбор? 1
длина ключей RSA может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072) 4096
Запрошенный размер ключа - 4096 бит
Выберите срок действия ключа.
  0 = не ограничен
  <n> = срок действия ключа - n дней
  <n>w = срок действия ключа - n недель
  <n>m = срок действия ключа - n месяцев
  <n>y = срок действия ключа - n лет
Срок действия ключа? (0) 0
Срок действия ключа не ограничен
```

Рис. 3.11: .

3.5 Настройка github

Создайте учётную запись на github.com. (рис. [3.12])

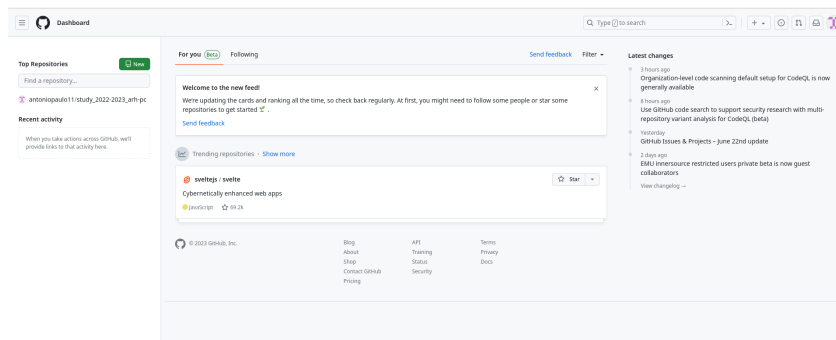


Рис. 3.12: .

Заполните основные данные на github.com. (рис. [3.13])

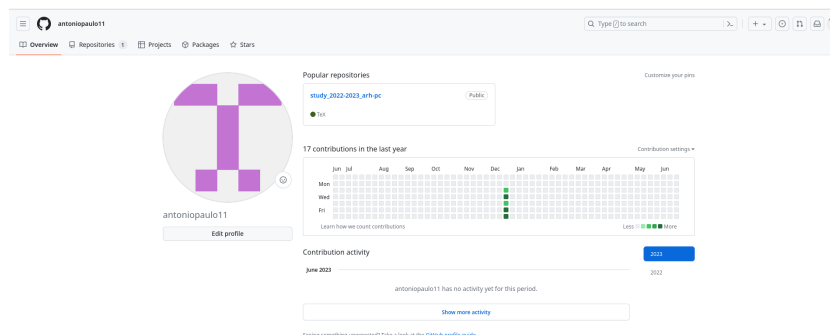


Рис. 3.13: .

3.6 Добавление PGP ключа в GitHub

Вывели список ключей и копировали отпечаток приватного ключа: (рис. [3.14])
Отпечаток ключа — это последовательность байтов, используемая для идентификации более длинного, по сравнению с самим отпечатком ключа.

```
[azpaulu@fedora ~]$ gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: глубина: 0 достоверных: 1 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f, 1u
/home/azpaulu/.gnupg/pubring.kbx
-----
sec   rsa4096/CA2CBAAB5EA03776 2023-06-23 [SC]
      FF8ECD114B3B8454C2610DF4CA2CBAAB5EA03776
uid   [ абсолютно ] Antonio Paulo <antoniopaulo33@icloud.com>
ssb   rsa4096/155E1772F08348D1 2023-06-23 [E]
```

Рис. 3.14: .

Скопировали сгенерированный PGP ключ в буфер обмена: (рис. [3.15])

```
[azpaulu@fedora ~]$ gpg --armor --export | xclip -sel clip
```

Рис. 3.15: .

Перешли в настройки GitHub, нажали на кнопку New GPG key и вставили полученный ключ в поле ввода. (рис. [3.16], [3.17])

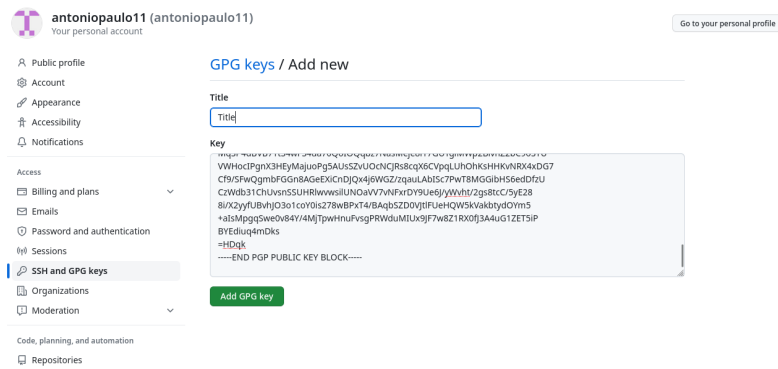


Рис. 3.16: .

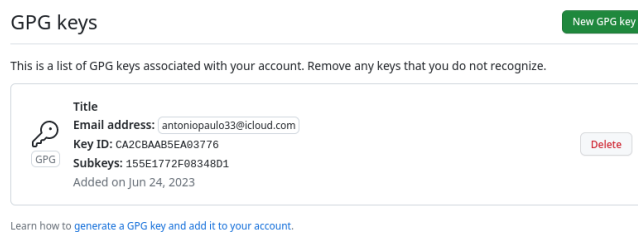


Рис. 3.17: .

3.7 Настройка автоматических подписей коммитов git

Используя введённый email, указали Git применять его при подписи коммитов: (рис. [3.18])

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

Рис. 3.18: .

3.8 Настройка gh

Авторизовались в gh. (рис. [3.19]) Утилита задали несколько наводящих вопросов.

```
[azpaulu@fedora ~]$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? SSH
? Upload your SSH public key to your GitHub account? /home/azpaulu/.ssh/id_rsa.pub
? Title for your SSH key: Title
? How would you like to authenticate GitHub CLI? Login with a web browser
```

Рис. 3.19: .

3.9 Создание репозитория курса на основе шаблона

Создали шаблон рабочего пространства. (рис. [3.20], [3.21], [3.22])

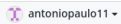
```
[azpaulu@fedora ~]$ mkdir -p ~/work/study/2022-2023/"Операционные системы"
[azpaulu@fedora ~]$ cd ~/work/study/2022-2023/"Операционные системы"
[azpaulu@fedora Операционные системы]$
```

Рис. 3.20: .

Create a new repository from course-directory-student-template

The new repository will start with the same files and folders as [yamadharma/course-directory-student-template](#).

Owner ⁺ Repository name ⁺

 antoniopaulo11 / study_2022-2023_os-intro ✓

Great repository names are short and memorable. Need inspiration? How about [potential-tribble](#)?

Description (optional)

☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

☐ **Include all branches**
Copy all branches from yamadharma/course-directory-student-template and not just master.

① You are creating a public repository in your personal account.

[Create repository from template](#)

Рис. 3.21: .

```
[azpaulu@fedora Операционные системы]$ git clone --recursive git@github.com:antonioapaulo
11/study_2022-2023_os-intro.git
Клонирование в «study_2022-2023_os-intro»...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 27 (delta 1), reused 11 (delta 0), pack-reused 0
Получение объектов: 100% (27/27), 16.93 КиБ | 16.93 МБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-
arkdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-te
mplate.git) зарегистрирован по пути «template/report»
Клонирование в «/home/azpaulu/work/study/2022-2023/Операционные системы/study_2022-2023_
os-intro/template/presentation»...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Получение объектов: 100% (82/82), 92.90 КиБ | 1.04 МБ/с, готово.
Определение изменений: 100% (28/28), готово.
Клонирование в «/home/azpaulu/work/study/2022-2023/Операционные системы/study_2022-2023_
os-intro/template/report»...
remote: Enumerating objects: 101, done.
```

Рис. 3.22: .

3.10 Настройка каталога курса

Перешли в каталог курса: (рис. [3.23])

```
azpaulu@fedora Операционные системы$ cd ~/work/study/2022-2023/"Операционные системы"/
study_2022-2023_os-intro
[azpaulu@fedora study_2022-2023_os-intro]$
```

Рис. 3.23: .

Удалили лишние файлы: (рис. [3.24])

```
study_2022-2023_os-intro
[azpaulu@fedora study_2022-2023_os-intro]$ rm package.json
[azpaulu@fedora study_2022-2023_os-intro]$
```

Рис. 3.24: .

Создали необходимые каталоги: (рис. [3.25])

```
[azpaulu@fedora study_2022-2023_os-intro]$ echo os-intro > COURSE
[azpaulu@fedora study_2022-2023_os-intro]$ make
```

Рис. 3.25: .

Отправили файлы на сервер: (рис. [3.26], [3.27])

```
[azpaulu@fedora study_2022-2023_os-intro]$ git add .  
[azpaulu@fedora study_2022-2023_os-intro]$ git commit -am 'feat(main): make course struc
```

Рис. 3.26: .

```
[azpaulu@fedora study_2022-2023_os-intro]$ git push
```

Рис. 3.27: .

4 Выводы

В ходе выполнения данной лабораторной работы была изучена идеология и применение средств контроля версий и освоены умения по работе с git.

5 Ответы на контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются? Система управления версиями (также используется определение «система контроля версий», от англ. Version Control System, VCS или Revision Control System) — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями 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 репозиторий, — место хранения файлов и их версий, служебной информации. Версия (revision), или ревизия, — состояние всего хранилища или отдельных файлов в момент времени («пункт истории»). Commit («трудовой вклад», не переводится) — процесс создания новой версии; иногда синоним версии. 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. Децентрализованные VCS: У каждого пользователя свой вариант (возможно не один) репозитория. Присутствует возможность добавлять и забирать изменения из любого репозитория (Git, Mercurial, Bazaar).

Централизованные VCS : Одно основное хранилище всего проекта Каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно (Subversion, CVS, TFS, VAULT, AccuRev)

6. Каковы основные задачи, решаемые инструментальным средством git? 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 init - создание репозитория git add (имена файлов) - Добавляет файлы в индекс git commit – выполняет коммит проиндексированных файлов в репозиторий git status – показывает какие файлы изменились между текущей стадией и HEAD. Файлы разделяются на 3 категории: новые файлы, измененные файлы, добавленные новые файлы git checkout (sha1 или метка) - получение указанной версии файла git push – отправка изменений в удаленный репозиторий git fetch – получение изменений из удаленного репозитория git clone (remote url) - клонирование удаленного репозитория себе
8. Приведите примеры использования при работе с локальным и удалённым репозиториями. (рис. [5.1])

```
azpaulu@fedora study_2022-2023_os-intro]$ git add .  
azpaulu@fedora study_2022-2023_os-intro]$ git commit -am 'feat(main): make course struc
```

Рис. 5.1: .

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 репозитории `git branch`. Создать ветку `git branch` имя.

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. Вот некоторые распространенные примеры таких файлов:

кэши зависимостей, например содержимое `node_modules` или `packages`; скомпилированный код, например файлы `.o`, `.рус` и `.class` ; каталоги для выходных данных сборки, например `bin`, `out` или `target`; файлы, сгенерированные во время выполнения, например `.log`, `.lock` или `.tmp`; скрытые системные файлы, например `.DS_Store` или `Thumbs.db`; личные файлы конфигурации IDE, например `.idea.workspace.xml`.