

Отчёт по лабораторной работе № 2

Операционные системы

Рыжов Егор

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	8
3.4	Создали ключи pgr	9
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	13
3.9	Создание репозитория курса на основе шаблона	13
3.10	Настройка каталога курса	14
4	Выводы	15
5	Ответы на контрольные вопросы	16

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	14
3.27	14
5.1	17
5.2	18
5.3	19

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])

```
[earihzhov@fedora ~]$ sudo -i
[sudo] пароль для earihzhov:
[root@fedora ~]# dnf install git
Fedora 36 - x86_64                               39 kB/s | 18 kB    00:00
Fedora 36 openh264 (From Cisco) - x86_64       1.8 kB/s | 989 B   00:00
Fedora Modular 36 - x86_64                     25 kB/s | 18 kB    00:00
Fedora 36 - x86_64 - Updates                   15 kB/s | 14 kB    00:00
Fedora 36 - x86_64 - [ === ] --- B/s | 0 B     --:-- ETA
```

Рис. 3.1: .

Установили gh:(рис. [3.2])

```
выполнено:
[root@fedora ~]# dnf install gh
Последняя проверка окончания срока действия метаданных: 0:00:20 назад, Сб 18 фев 2023 17:03:36.
Зависимости разрешены.
=====
Пакет      Архитектура  Версия      Репозиторий  Размер
=====
Установка:
gh          x86_64       2.23.0-1.fc36 updates      8.2 МБ
=====
Результат транзакции
=====
Установка 1 Пакет

Объем загрузки: 8.2 М
Объем изменений: 41 М
Продолжить? [д/Н]:
```

Рис. 3.2: .

3.2 Базовая настройка git

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

```
[root@fedora ~]# git config --global user.name "Egor Rizhov"
[root@fedora ~]# git config --global user.email "ao12121@yandex.ru"
[root@fedora ~]#
```

Рис. 3.3: .

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

```
[earihzhov@fedora ~]$ git config --global core.quotepath false
[earihzhov@fedora ~]$
```

Рис. 3.4: .

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

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

Рис. 3.5: .

Параметр autocrlf:(рис. [3.6])

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

Рис. 3.6: .

Параметр safecrlf: (рис. [3.7])

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

Рис. 3.7: .

3.3 Создали ключи ssh

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

```
[earihzhov@fedora ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.

Enter file in which to save the key (/home/earihzhov/.ssh/id_rsa): /home/earihzhov/.ssh/id_rsa
v/.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/earihzhov/.ssh/id_rsa
Your public key has been saved in /home/earihzhov/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:aFwWEGRu9k0INz0wXQ/xTmHbdgCufScUk6APYeYJZzc earihzhov@fedora
The key's randomart image is:
+---[RSA 4096]-----+
|
|..oB+..E+..|
|o o %*+ *.B|
|+ + *.. * +|
|+ = o = + ..|
|+ S o o + .|
|..      . o |
|              |
|              |
+-----[SHA256]-----+
```

Рис. 3.8: .

по алгоритму ed25519: (рис. [3.9])

```
[earihzhov@fedora ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/earihzhov/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/earihzhov/.ssh/id_ed25519
Your public key has been saved in /home/earihzhov/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:kQja702YF6N6diqZ76k8ooNdfWaPkcCfDyKfVh9uqQU earihzhov@fedora
The key's randomart image is:
+--[ED25519 256]--+
|
|+ . . .
|. o.. o
|..o .
|..+oEo
|o=oo@..
|.. ++ooB O.o
|+*.oo= ..B
|oo+0=+ .o
+-----[SHA256]-----+
```

Рис. 3.9: .

3.4 Создали ключи ргр

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])

```
[earihzhov@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/earihzhov/.gnupg'
gpg: создан шит с ключами '/home/earihzhov/.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
Все верно? (y/N) y
```

Рис. 3.10: .

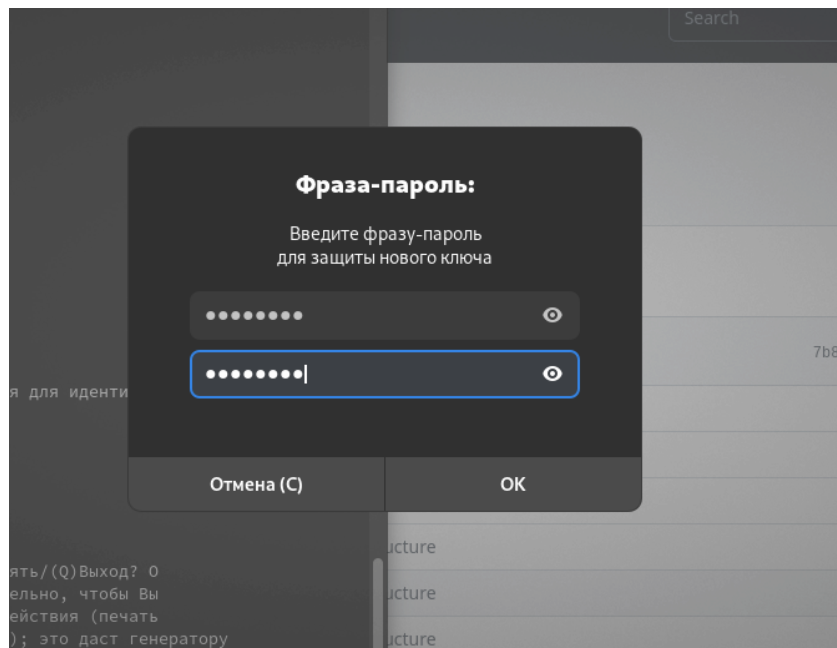


Рис. 3.11: .

3.5 Настройка github

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

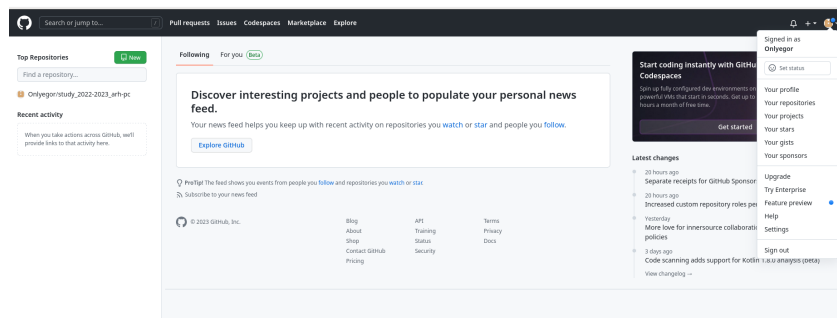


Рис. 3.12: .

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

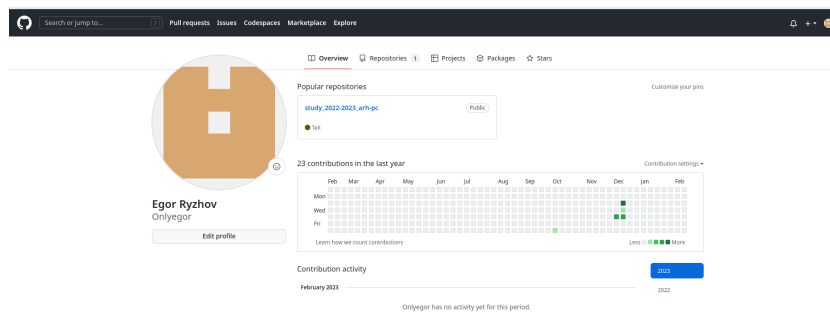


Рис. 3.13: .

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.

```
[earihzhov@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/earihzhov/.gnupg/pubring.kbx
-----
sec   rsa4096/F1629FEE7849724D 2023-02-18 [SC]
      EC1E249559191797AED811F1629FEE7849724D
uid   [ абсолютно ] Egor Rizhov (Tub) <ao12121@yandex.ru>
ssb   rsa4096/82A204B5FA24843D 2023-02-18 [E]
```

Рис. 3.14: .

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

```
[earihzhov@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])

GPG keys / Add new

Title

Title

Key

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
mQINB/kSvA3ZDMLMTd+ta9c3W5r5PqSGQrq/oQqsGvmVTTrVmGfMrwgX93UsKS0zCn
MGgTK1jN+I2pw2qIcoTh3oNTuvUjhPTzeLXBznGqg7DCBRWTq74IU3O0yYmncIpy
dsiXe7qYGwgiBB+14CtH4Y5aIRFRpDSR1S4oTC4GLbT8d5l1Lh0SxVtnl93YsUaI
eQu0v6bCewliMJU1D+Ph1HT7bvFbjOE1fyTO4qSCoTyTFX+cXusFFG2PX41jB16t
C0jB1spPHKjXceiqTJpcUYdCsyEof5Y2TGoAdBgkHvArjUxcTu2FJtg++0yx56oV
OizOWQQ=
=0eS6
-----END PGP PUBLIC KEY BLOCK-----
```


Add GPG key

Рис. 3.16: .

GPG keys

New GPG key

This is a list of GPG keys associated with your account. Remove any keys that you do not recognize.

 **Title**

Email address: ao12121@yandex.ru

Key ID: F1629FEE7849724D

Subkeys: 82A204B5FA24843D

Added on Feb 18, 2023

Delete

[Learn how to generate a GPG key and add it to your account.](#)

Рис. 3.17: .

3.7 Настройка автоматических подписей коммитов git

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

```
[earihzhov@fedora ~]$ git config --global user.signingkey
[earihzhov@fedora ~]$ git config --global commit.gpgsign true
[earihzhov@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.


```
[earihzhov@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/earihzhov/.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])


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

Owner * / Repository name *
 Onlyegor / ✓


Great repository names [study_2022-2023_os-intro](#) is available. Inspiration? How about [studious-octo-waddle](#)?

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 yamadharm/course-directory-student-template and not just master.

 You are creating a public repository in your personal account.

[Create repository from template](#)

Рис. 3.20: .

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

Рис. 3.21: .

```
[earihzhov@fedora Операционные системы]$ git clone --recursive git@github.com:Onlyegor/study_2022-2023_os-intro.git
Клонирование в «study_2022-2023_os-intro»...
```

Рис. 3.22: .

3.10 Настройка каталога курса

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

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

Рис. 3.23: .

Удалили лишние файлы: (рис. [3.24])

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

Рис. 3.24: .

Создали необходимые каталоги: (рис. [3.25])

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

Рис. 3.25: .

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

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

Рис. 3.26: .

```
[earihzhov@fedora Операционные системы]$ 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)

4. Опишите действия с VCS при единоличной работе с хранилищем. (рис. [5.1])

Единоличная работа с VCS

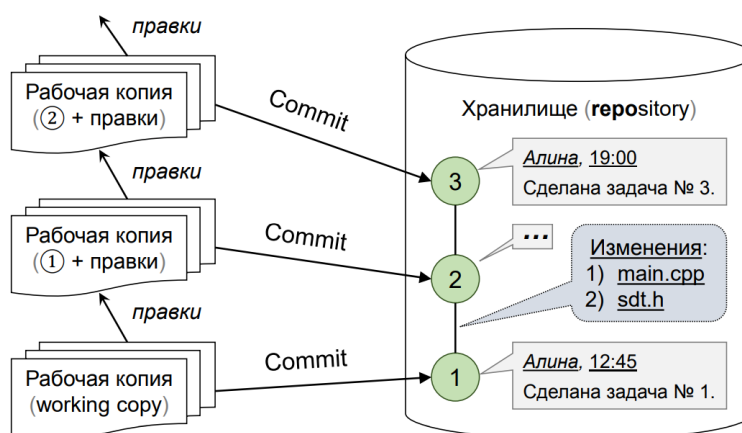


Рис. 5.1: .

5. Опишите порядок работы с общим хранилищем VCS. (рис. [5.2])

Работа с общим хранилищем

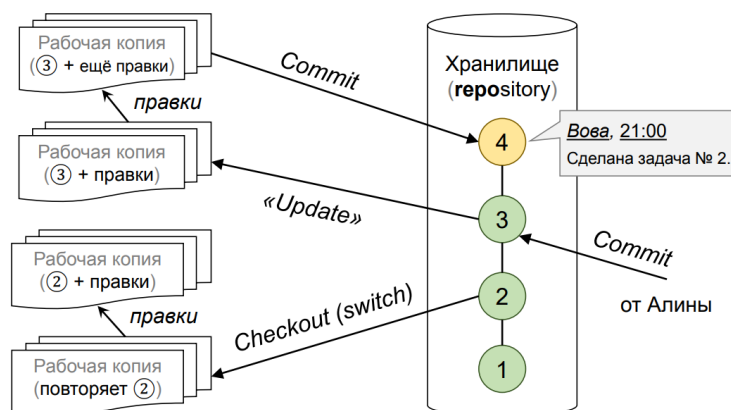


Рис. 5.2: .

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. [5.3])

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

Рис. 5.3: .

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.