

Лабораторная работа №2

Первоначальная настройка git

Дисциплина Операционные системы

Фадин В.В. НПМбв-02-20

Содержание

1 Цель работы	5
2 Выполнение лабораторной работы	6
2.1 Настройка GIT	6
2.2 Создание рабочего пространства	9
3 Ответы на контрольные вопросы	11
4 Выводы	14

Список иллюстраций

2.1	Создание PGP ключа	8
2.2	Авторизация в Github	9
2.3	Создание репозитория 1	9
2.4	Удаление файла package.json и создание файла COURSE	10
2.5	Git push	10

Список таблиц

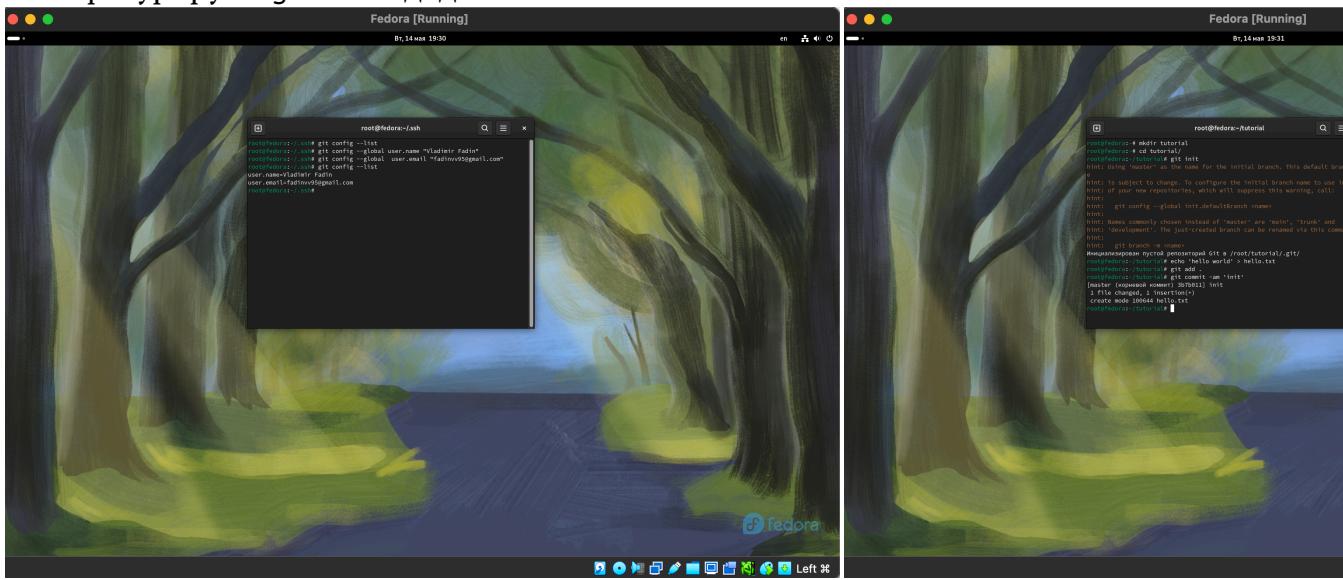
1 Цель работы

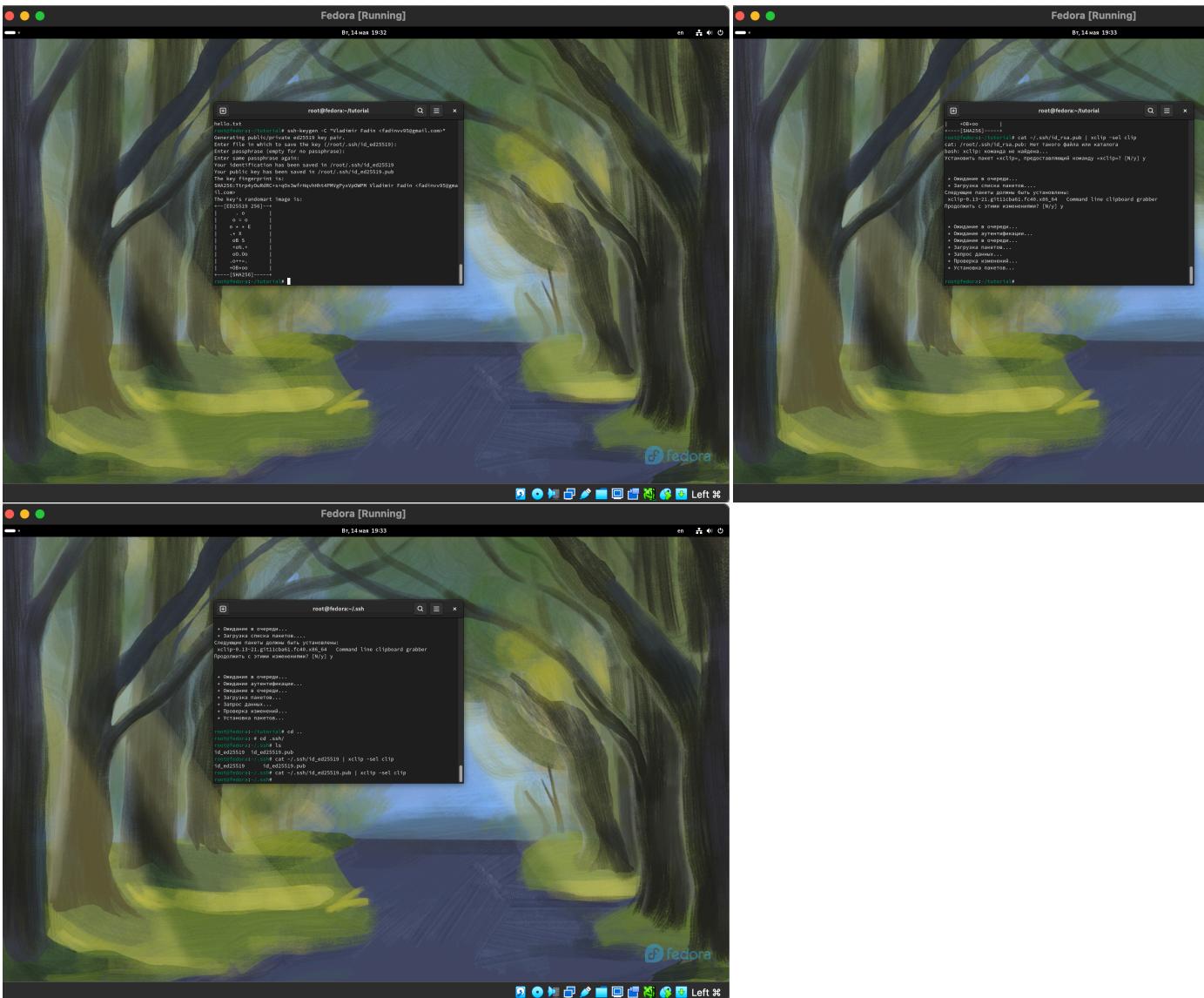
- Изучить применение средств контроля версий
- Освоить умения по работе с git'ом

2 Выполнение лабораторной работы

2.1 Настройка GIT

Сконфигурируем git и создадим SSH ключ





Создадим PGP ключ

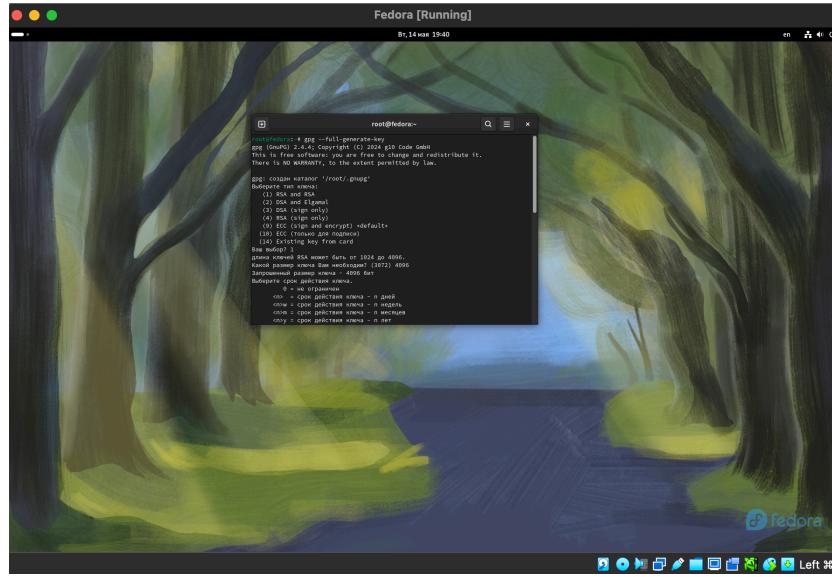


Рис. 2.1: Создание PGP ключа

С помощью команды `gpg --armor --export <PGP Fingerprint> | xclip -sel clip` скопируем PGP ключ. И вставим в наш Github.

Владимир Фадин (Fadinv)
Your personal account

- [Public profile](#)
- [Account](#)
- [Appearance](#)
- [Accessibility](#)
- [Notifications](#)
- [Access](#)
 - [Billing and plans](#)
 - [Emails](#)
 - [Password and authentication](#)
 - [Sessions](#)
 - SSH and GPG keys**
 - [Organizations](#)
 - [Enterprises](#)
 - [Moderation](#)
- [Code, planning, and automation](#)
 - [Repositories](#)
 - [Packages](#)
 - [Copilot](#)
 - [Pages](#)
 - [Saved replies](#)
- [Security](#)
- [Code security and analysis](#)
- [Integrations](#)
- [Applications](#)
- [Scheduled reminders](#)
- [Archives](#)
 - [Security log](#)
 - [Sponsorship log](#)
- [Developer settings](#)

SSH keys

This is a list of SSH keys associated with your account. Remove any keys you no longer need.

Key	Added	Actions
macbook m1	Added on May 14, 2024	SSH

Check out our guide to [connecting to GitHub using SSH keys](#) or [troubleshoot common issues](#).

GPG keys

This is a list of GPG keys associated with your account. Remove any keys you no longer need.

Key	Email address	Key ID	Subkeys	Added	Actions
os_fadin	fadinv@gmail.com	67682F2E3088C8F888	SOC00525B53B3EDF	Added on May 14, 2024	View

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

Vigilant mode

Flag unsigned commits as unverified

Авторизуемся в Github

8

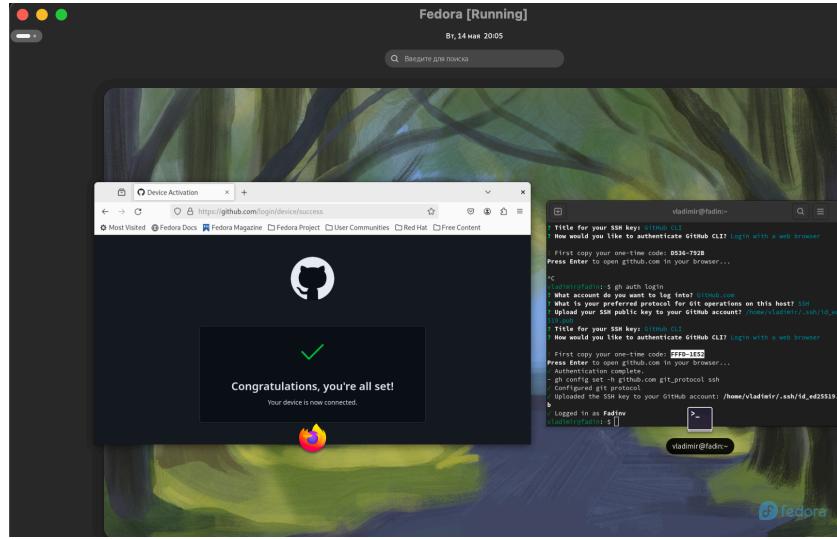


Рис. 2.2: Авторизация в Github

2.2 Создание рабочего пространства

Переходим в репозиторий с шаблоном и создаем из него шаблон.

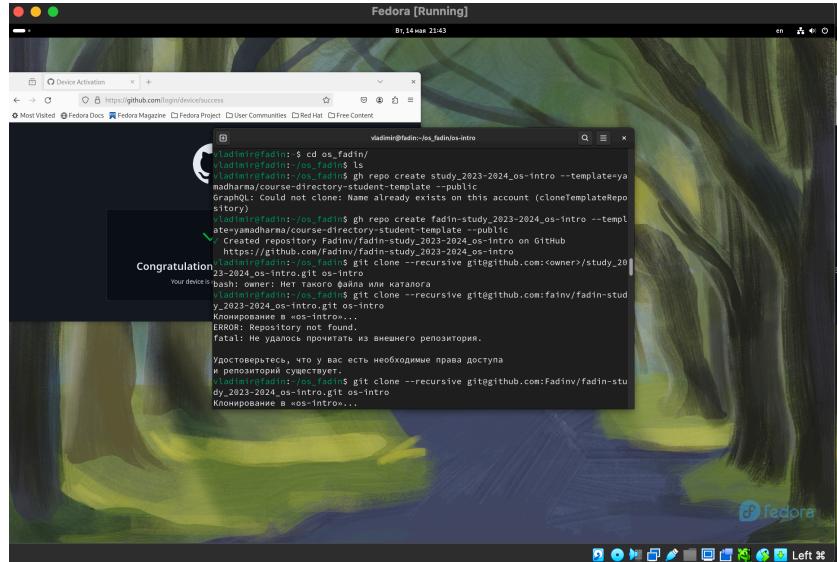


Рис. 2.3: Создание репозитория 1

Перейдем в репозиторий и удалим package.json. Также создадим файл COURSE с текстом “os-intro”

```

Fedora [Running]
B:14 мэр 21:43

Device Activation | https://github.com/login/device/success
Most Visited | Fedora Docs | Fedora Magazine | Fedora Project | User Communities | Red Hat | Free Content

vladimir@vladimir-laptop:~/os_fadin/os-intro$ rm package.json
vladimir@vladimir-laptop:~/os_fadin/os-intro$ echo os-intro > COURSE
vladimir@vladimir-laptop:~/os_fadin/os-intro$ make
Usage:
  make <target>

Targets:
  list           List of courses
  prepare        Generate directories structure
  submodule      Update submodules

Congratulations!
Your device is now ready to use.

  LICENSE  package.json  README.git-flow.md  template
  CHANGELOG.md  COURSE  Makefile  README.git-flow.md  template
  LICENSE  README.en.md  README.md
vladimir@vladimir-laptop:~/os_fadin/os-intro$ ls
CHANGELOG.md  COURSE  Makefile  README.en.md  README.md
LICENSE  README.git-flow.md  template
README.git-flow.md
vladimir@vladimir-laptop:~/os_fadin/os-intro$ ls
CHANGELOG.md  LICENSE  Makefile  README.en.md  README.git-flow.md
COURSE  Makefile  project-personal  README.git-flow.md
README.git-flow.md
vladimir@vladimir-laptop:~/os_fadin/os-intro$ cd labs/
vladimir@vladimir-laptop:~/os_fadin/os-intro/labs$ ls
lab01  lab02  lab03  lab04  lab05  lab06  lab07  lab08  README.md
vladimir@vladimir-laptop:~/os_fadin/os-intro/labs$ git add .
vladimir@vladimir-laptop:~/os_fadin/os-intro/labs$ git commit -m "make course structure"
[lab08 749d454] make course structure
  1 file changed, 1 insertion(+)
vladimir@vladimir-laptop:~/os_fadin/os-intro$ git push
git: 'push' is not a git command. See 'git --help'.
vladimir@vladimir-laptop:~/os_fadin/os-intro$ git push
Counting objects: 40, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (38/38), done.
Writing objects: 100% (38/38), 342.23 KiB | 2.55 MiB/c, done.
Total 38 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0).
remote: Resolving deltas: 100% (4/4) completed with 1 local object.
To https://github.com/vladimir-fadin/os-intro.git
   749d454..207d45d master -> master
vladimir@vladimir-laptop:~/os_fadin/os-intro$
```

Рис. 2.4: Удаление файла package.json и создание файла COURSE

Теперь запушим изменения. До этого исполнив git add и git commit с назначением коммита feat(main): make course structure

```

Fedora [Running]
B:14 мэр 21:43

Device Activation | https://github.com/login/device/success
Most Visited | Fedora Docs | Fedora Magazine | Fedora Project | User Communities | Red Hat | Free Content

vladimir@vladimir-laptop:~/os_fadin/os-intro$ rm package.json
vladimir@vladimir-laptop:~/os_fadin/os-intro$ echo os-intro > COURSE
vladimir@vladimir-laptop:~/os_fadin/os-intro$ make
Usage:
  make <target>

Targets:
  list           List of courses
  prepare        Generate directories structure
  submodule      Update submodules

Congratulations!
Your device is now ready to use.

  LICENSE  package.json  README.git-flow.md  template
  CHANGELOG.md  COURSE  Makefile  README.git-flow.md  template
  LICENSE  README.en.md  README.md
vladimir@vladimir-laptop:~/os_fadin/os-intro$ ls
CHANGELOG.md  COURSE  Makefile  README.en.md  README.md
LICENSE  README.git-flow.md  template
README.git-flow.md
vladimir@vladimir-laptop:~/os_fadin/os-intro$ ls
CHANGELOG.md  LICENSE  Makefile  README.en.md  README.git-flow.md
COURSE  Makefile  project-personal  README.git-flow.md
README.git-flow.md
vladimir@vladimir-laptop:~/os_fadin/os-intro$ cd labs/
vladimir@vladimir-laptop:~/os_fadin/os-intro/labs$ ls
lab01  lab02  lab03  lab04  lab05  lab06  lab07  lab08  README.md
vladimir@vladimir-laptop:~/os_fadin/os-intro/labs$ git add .
vladimir@vladimir-laptop:~/os_fadin/os-intro/labs$ git commit -m "make course structure"
[lab08 749d454] make course structure
  1 file changed, 1 insertion(+)
vladimir@vladimir-laptop:~/os_fadin/os-intro$ git push
git: 'push' is not a git command. See 'git --help'.
vladimir@vladimir-laptop:~/os_fadin/os-intro$ git push
Counting objects: 40, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (38/38), done.
Writing objects: 100% (38/38), 342.23 KiB | 2.55 MiB/c, done.
Total 38 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0).
remote: Resolving deltas: 100% (4/4) completed with 1 local object.
To https://github.com/vladimir-fadin/os-intro.git
   749d454..207d45d master -> master
vladimir@vladimir-laptop:~/os_fadin/os-intro$
```

Рис. 2.5: Git push

3 Ответы на контрольные вопросы

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

Системы контроля версий (VCS) предназначены для отслеживания изменений в программном коде и обеспечения коллективной разработки.

3.0.0.2 Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.

Хранилище: Место, где хранятся все изменения и версии программного кода.
Commit: Отдельное изменение или набор изменений в коде, зафиксированное в системе контроля версий. История: Последовательность коммитов, отображающая эволюцию кода. Рабочая копия: Локальная копия проекта, с которой работает разработчик.

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

Децентрализованные VCS копируют всю историю изменений на каждый клиентский компьютер, в то время как централизованные VCS хранят все изменения на центральном сервере и клиенты получают только последние версии файлов. Примеры децентрализованных VCS: Git, Mercurial. Примеры централизованных VCS: Subversion, CVS.

3.0.0.4 Опишите действия с VCS при единоличной работе с хранилищем.

При индивидуальной разработке пользователь клонирует проект на свой компьютер, вносит изменения и создает новые версии, коммитя их в системе контроля версий.

3.0.0.5 Опишите порядок работы с общим хранилищем VCS.

Пользователь получает версию проекта из центрального хранилища, вносит изменения, коммитит их и отправляет обратно в хранилище.

3.0.0.6 Каковы основные задачи, решаемые инструментальным средством git?

Git используется для разработки проектов в команде, контроля изменений в файлах и возможности сохранения нескольких состояний проекта.

3.0.0.7 Назовите и дайте краткую характеристику командам git.

`git add` - добавляет изменения для коммита.

`git commit` - сохраняет изменения в репозитории с названием.

`git push` - отправляет изменения на удаленный репозиторий.

`git config` - позволяет изменить настройки Git.

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

В локальном репозитории разработчик может вносить изменения в код и коммитить их без доступа к сети. В удаленном репозитории команда разработчиков может совместно работать над проектом, обмениваясь изменениями через централизованный сервер.

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

Ветви используются для параллельной разработки функций или исправлений, чтобы избежать конфликтов между изменениями и обеспечить безопасное тестирование нового кода.

3.0.0.10 Как и зачем можно игнорировать некоторые файлы при commit?

Файлы могут быть проигнорированы с помощью файла .gitignore, чтобы избежать загрязнения репозитория лишними или конфиденциальными файлами.

4 Выводы

Мы изучили идеологию применения средств контроля версий и освоили базовые команды git'a.