

Лабораторная работа № 2.

Операционные системы

Шулуужук Айраана Вячеславовна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Установка программного обеспечения	7
3.2	Базовая настройка git	7
3.3	Создание ключей ssh	8
3.4	Создание pgr ключа	9
3.5	Создание репозитория курса на основе шаблона	10
3.6	Настройка каталога курса	11
4	Ответы на контрольные вопросы	13
5	Выводы	16

Список иллюстраций

3.1	Установка программного обеспечения	7
3.2	Базовая настройка git	8
3.3	генерация ключей	8
3.4	созданный ключ в гит	9
3.5	ключ pgr	9
3.6	добавление ключа в гит	10
3.7	клонирование репозитория	10
3.8	созданный репозитрий	10
3.9	Настройка каталога курса	11
3.10	отправление файлов на сервер	11

Список таблиц

1 Цель работы

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

2 Задание

1. Создать базовую конфигурацию для работы с git.
2. Создать ключ SSH.
3. Создать ключ PGP.
4. Настроить подписи git.
5. Зарегистрироваться на Github.
6. Создать локальный каталог для выполнения заданий по предмету.

3 Выполнение лабораторной работы

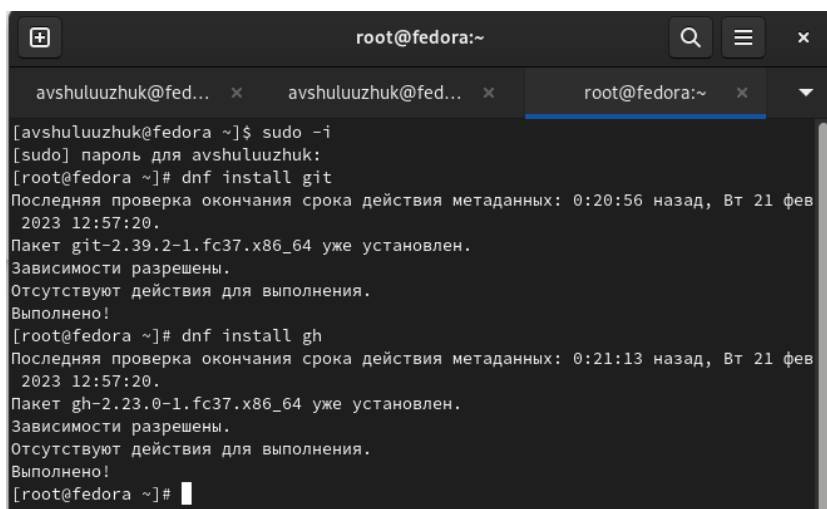
3.1 Установка программного обеспечения

Установка git (рис. 3.1):

```
dnf install git
```

Установка gh:

```
dnf install gh
```



```
root@fedora:~  
[avshuluuzhuk@fedora ~]$ sudo -i  
[sudo] пароль для avshuluuzhuk:  
[root@fedora ~]# dnf install git  
Последняя проверка окончания срока действия метаданных: 0:20:56 назад, Вт 21 фев 2023 12:57:20.  
Пакет git-2.39.2-1.fc37.x86_64 уже установлен.  
Зависимости разрешены.  
Отсутствуют действия для выполнения.  
Выполнено!  
[root@fedora ~]# dnf install gh  
Последняя проверка окончания срока действия метаданных: 0:21:13 назад, Вт 21 фев 2023 12:57:20.  
Пакет gh-2.23.0-1.fc37.x86_64 уже установлен.  
Зависимости разрешены.  
Отсутствуют действия для выполнения.  
Выполнено!  
[root@fedora ~]#
```

Рис. 3.1: Установка программного обеспечения

3.2 Базовая настройка git

Продельываем предварительную конфигурацию гит (рис. 3.2)

```
avshuluuzhuk@fedora:~  
[avshuluuzhuk@fedora ~]$ git config --global user.name "<avshuluuzhuk>"  
[avshuluuzhuk@fedora ~]$ git config --global user.email "<1132221890@pfur.ru>"  
[avshuluuzhuk@fedora ~]$ git config --global core.quotepath false  
[avshuluuzhuk@fedora ~]$ git config --global init.defaultBranch master  
[avshuluuzhuk@fedora ~]$ git config --global core.autocrlf input  
[avshuluuzhuk@fedora ~]$ git config --global core.safecrlf warn  
[avshuluuzhuk@fedora ~]$
```

Рис. 3.2: Базовая настройка git

3.3 Создание ключей ssh

Создаем ключи (рис. 3.3)

по алгоритму rsa с ключём размером 4096 бит:

```
ssh-keygen -t rsa -b 4096
```

по алгоритму ed25519:

```
ssh-keygen -t ed25519
```

```
[avshuluuzhuk@fedora ~]$ ssh-keygen -C "avshuluuzhuk <1132221890@pfur.ru>"  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/avshuluuzhuk/.ssh/id_rsa):  
Created directory '/home/avshuluuzhuk/.ssh'.  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/avshuluuzhuk/.ssh/id_rsa  
Your public key has been saved in /home/avshuluuzhuk/.ssh/id_rsa.pub  
The key fingerprint is:  
SHA256:FuVAtf8VptW8R7/5XkXxBdZCXYLHPoZDbzLDY0YqfIA avshuluuzhuk <1132221890@pfur.ru>  
The key's randomart image is:  
+---[RSA 3072]-----+  
|      .oo+ .+B+o|  
|      E= o.=B+B|  
|      .=.oB BB|  
|      .+o &o+|  
|      $ ..=|  
|      . .|=|  
|      .O.|  
|      o|  
|      .o|  
+---[SHA256]-----+
```

Рис. 3.3: генерация ключей

Загрузим ключи в гит(рис. 3.4), скопируем их, используя команду :


```
xclip -i < ~/.ssh/id_ed25519.pub
```

SSH keys

[New SSH key](#)

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

Authentication Keys



	title SHA256:Hk76davidYP+yqAvM8WriNR9nw3r+G1SLyqNdUwv4ky8 Added on Nov 16, 2022 Last used within the last 4 months — Read/write	Delete
	rsa4096 SHA256:tkyXLV6TuKnhPjBhASSYJq10Ekv3EBpGznkiImynyng Added on Feb 18, 2023 Last used within the last week — Read/write	Delete

Рис. 3.4: созданный ключ в гит

3.4 Создание pgp ключа

Генерируем ключ (рис. 3.5)

```
[avshuluuzhuk@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/avshuluuzhuk/.gnupg/pubring.kbx
-----
sec   rsa4096/E9A38B1B8BD87934 2023-02-21 [SC]
      592A9B6D1D56C790CBF29790E9A38B1B8BD87934
uid           [ абсолютно ] Shuluuzhuk Ayraana <1132221890@pfur.ru>
ssb   rsa4096/E454083A3FCB84B 2023-02-21 [E]
```

Рис. 3.5: ключ pgp


Скопируем этот ключ и добавляем в гит (рис. 3.6)

```
gpg --armor --export <PGP Fingerprint> | xclip -sel clip
```

GPG keys

[New GPG key](#)

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

 1
Email address: 1132221890@pfur.ru
Key ID: E9A38B1B8BD87934
Subkeys: E454083A3CFCB84B
Added on Feb 21, 2023
[Delete](#)

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

Рис. 3.6: добавление ключа в гит

3.5 Создание репозитория курса на основе шаблона

Используя репозиторий с шаблоном курса, создаем репозиторий “Операционные системы”. Клонировать созданный репозиторий (рис. 3.7), ссылку для клонирования копируем на странице созданного репозитория:

```
git clone --recursive
```

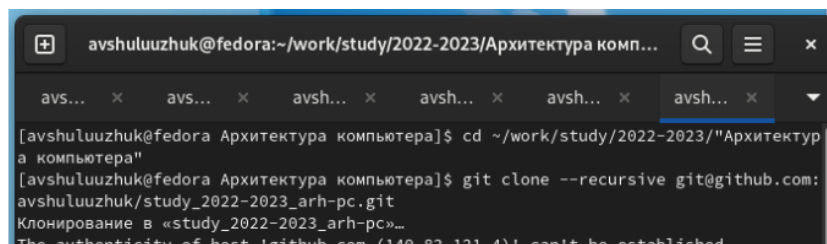


Рис. 3.7: клонирование репозитория

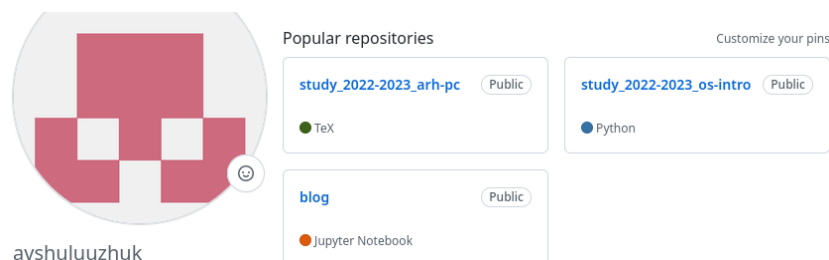
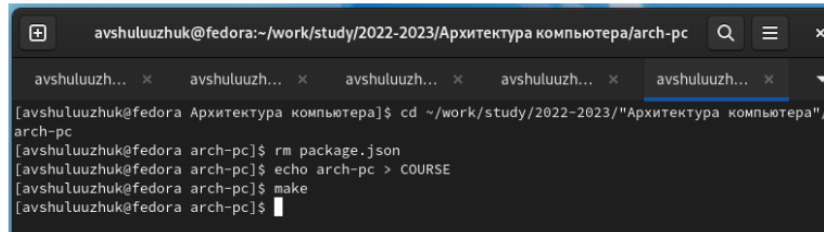


Рис. 3.8: созданный репозиторий

3.6 Настройка каталога курса

Переходим в каталог курса и удаляем лишние файлы (рис. 3.9)

```
rm package.json
```



```
avshuluuzhuk@fedora:~/work/study/2022-2023/Архитектура компьютера/arch-pc
[avshuluuzhuk@fedora Архитектура компьютера]$ cd ~/work/study/2022-2023/"Архитектура компьютера"/arch-pc
[avshuluuzhuk@fedora arch-pc]$ rm package.json
[avshuluuzhuk@fedora arch-pc]$ echo arch-pc > COURSE
[avshuluuzhuk@fedora arch-pc]$ make
[avshuluuzhuk@fedora arch-pc]$
```

Рис. 3.9: Настройка каталога курса

Создаем необходимые файлы:

```
echo os-intro > COURSE
```

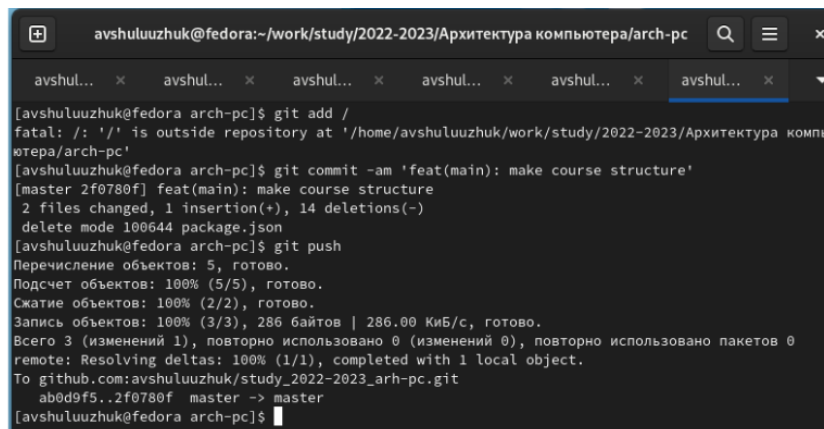
```
make
```

Отправляем файлы на сервер (рис. 3.10)

```
git add .
```

```
git commit -am 'feat(main): make course structure'
```

```
git push
```



```
avshuluuzhuk@fedora:~/work/study/2022-2023/Архитектура компьютера/arch-pc
[avshuluuzhuk@fedora arch-pc]$ git add /
fatal: /: '/' is outside repository at '/home/avshuluuzhuk/work/study/2022-2023/Архитектура компьютера/arch-pc'
[avshuluuzhuk@fedora arch-pc]$ git commit -am 'feat(main): make course structure'
[master 2f0780f] feat(main): make course structure
2 files changed, 1 insertion(+), 14 deletions(-)
delete mode 100644 package.json
[avshuluuzhuk@fedora arch-pc]$ git push
Перечисление объектов: 5, готово.
Подсчет объектов: 100% (5/5), готово.
Сжатие объектов: 100% (2/2), готово.
Запись объектов: 100% (3/3), 286 байтов | 286.00 КиБ/с, готово.
Всего 3 (изменений 1), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:avshuluuzhuk/study_2022-2023_arh-pc.git
 ab0d9f5..2f0780f master -> master
[avshuluuzhuk@fedora arch-pc]$
```

Рис. 3.10: отправление файлов на сервер

Далее проверяем правильность создания иерархии рабочего пространства в локальном репозитории и на странице гит

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 репозиторий, место хранения всех версий и служебной информации. Commit (¾[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 репозиториях 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 ветки (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

Были приобретены практические навыки по работе с системой git. Изучена идеология и применение средств контроля версий. В процессе лабораторной работы выполнено создание рабочего пространства и нового репозитория курса на основе шаблона, загрузка файлов на github