

Лабораторная работа №2

Дисциплина: Архитектура компьютера

Серебрякова Дарья Ильинична

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
4.1	Настройка github	9
4.2	Базовая настройка git	10
4.3	Создание SSH ключа	11
4.4	Создание рабочего пространства и репозитория курса на основе шаблона	13
4.5	Создание репозитория курса на основе шаблона	13
4.6	Настройка каталога курса	15
5	Вывод	16
	Список литературы	17

Список иллюстраций

4.1	Созданный аккаунт	10
4.2	Создание предварительной конфигурации	11
4.3	Настройка utf-8 в выводе сообщений git	11
4.4	Создание имени начальной ветки и параметров для нее	11
4.5	Сохранение ключа	12
4.6	Копия ключа в буфер обмена	12
4.7	Создание ключа	12
4.8	Ключ создан	13
4.9	Создание каталога для предмета	13
4.10	Создание репозитория	14
4.11	Переход в каталог курса	14
4.12	Ссылка для клонирования	14
4.13	Клонирования созданного репозитория	15
4.14	Переход в каталог курса	15
4.15	Удаление лишних файлов	15
4.16	Создание необходимых каталогов	15

Список таблиц

1 Цель работы

Целью работы является изучить идеологию и применение средств контроля версий. Приобрести практические навыки по работе с системой git

2 Задание

1. Настройка GitHub.
2. Базовая настройка Git.
3. Создание SSH-ключа.
4. Создание рабочего пространства и репозитория курса на основе шаблона.
5. Создание репозитория курса на основе шаблона.
6. Настройка каталога курса.
7. Выполнение заданий для самостоятельной работы.

3 Теоретическое введение

Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется. В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять не полную версию изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных. Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить изменения, сделанные разными участниками, вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зависимости от настроек блокировка не позволяет

другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом привилегированный доступ только одному пользователю, работающему с файлом. Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным. Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд. Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды `git` с различными опциями. Благодаря тому, что Git является распределённой системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией. Работа пользователя со своей веткой начинается с проверки и получения изменений из центрального репозитория (при этом в локальное дерево до начала этой процедуры не должно было вноситься изменений). Затем можно вносить изменения в локальное дерево и/или ветке. После завершения внесения какого-то изменения в файлы и/или каталоги проекта необходимо разместить их в центральном репозитории.

4 Выполнение лабораторной работы

4.1 Настройка github

Создала учётную запись на сайте <https://github.com/> и заполнила основные данные. Вошла в свой аккаунт (рис. 4.1):

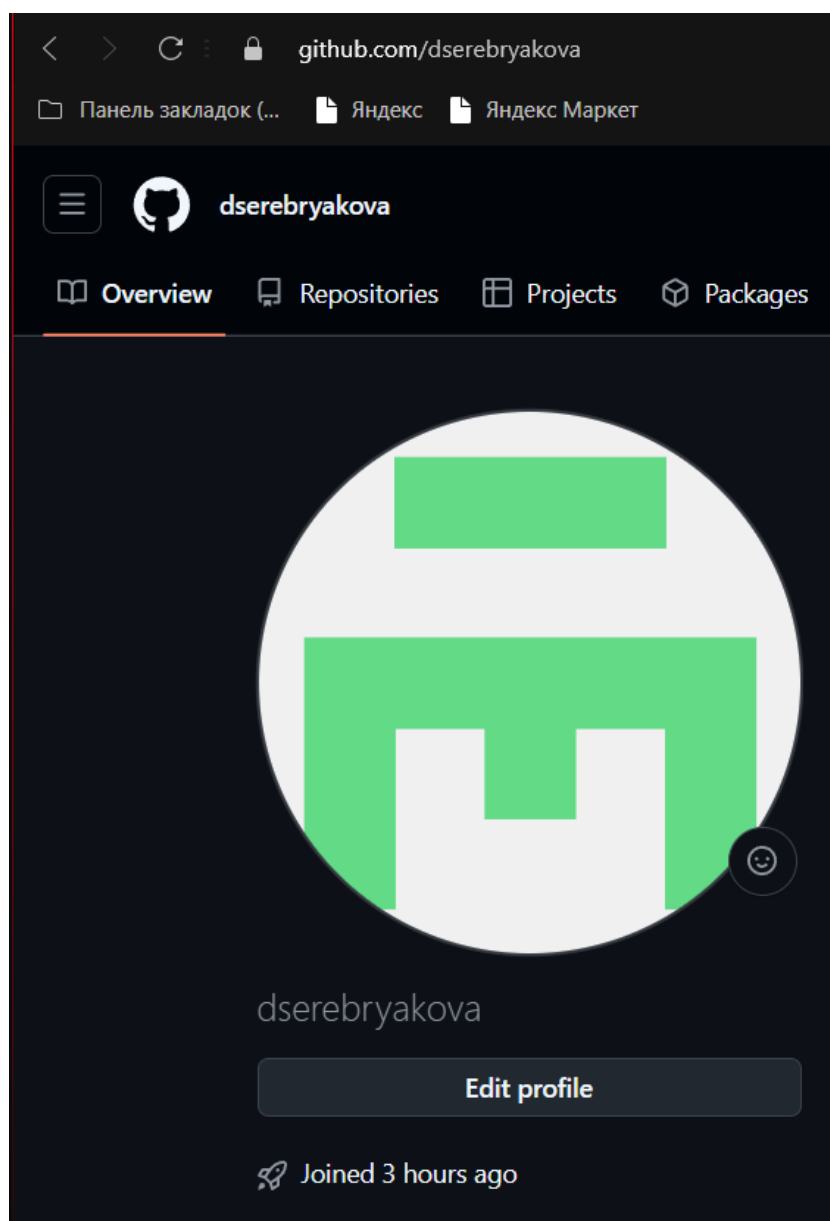


Рис. 4.1: Созданный аккаунт

4.2 Базовая настройка git

Открываю виртуальную машину, захожу в терминал. Сделаю предварительную конфигурацию git. Ввожу команду `git config --global user.name ""`, указывая свое имя, фамилию. Далее ввожу команду `git config --global user.email "work@mail"`, указывая в ней свою электронную почту (рис. 4.2):

```
diserebryakova@fedora:~$ git config --global user.name "<Дарья Серебрякова>"
diserebryakova@fedora:~$ git config --global user.email "<1132246733@pfur.ru>"
diserebryakova@fedora:~$
```

Рис. 4.2: Создание предварительной конфигурации

Настраиваю utf-8 в выводе сообщений git, введя предложенную команду (рис. 4.3):

```
diserebryakova@fedora:~$ git config --global core.quotePath false
diserebryakova@fedora:~$
```

Рис. 4.3: Настройка utf-8 в выводе сообщений git

Задаю имя начальной ветки (буду называть её masster). Также задаю параметр autocrlf со значением input и параметр safecrlf со значением warn (рис. 4.4):

```
diserebryakova@fedora:~$ git config --global init.defaultBranch masster
diserebryakova@fedora:~$ git config --global core.autocrlf input
diserebryakova@fedora:~$ git config --global core.safecrlf warn
diserebryakova@fedora:~$
```

Рис. 4.4: Создание имени начальной ветки и параметров для нее

4.3 Создание SSH ключа

Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый). Для этого ввожу команду `ssh-keygen -C "Имя Фамилия, work@email"`, указывая свое имя (имя владельца) и электронную почту владельца. Ключ автоматически сохранится в каталоге `~/.ssh/` (рис. 4.5):

```

diserebryakova@fedora: ~$ ssh-keygen -C "Дарья Серебрякова <1132246733@pfur.ru>"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/diserebryakova/.ssh/id_ed25519):
Created directory '/home/diserebryakova/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/diserebryakova/.ssh/id_ed25519
Your public key has been saved in /home/diserebryakova/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:qLBPT0gVHptsxCid/aGUBQ8J8dxUjfrHCBWDWJqrDXa4 Дарья Серебрякова <1132246733@pfur.ru>
The key's randomart image is:
+--[ED25519 256]--+
| .oB*+++++.. |
| .++*o+.o. |
| .X+o+.. |
| oo.o. |
| ...o.oS |
| ++o.. |
| .+... |
| o.oE |
| . |
+-----[SHA256]-----+
diserebryakova@fedora: ~$

```

Рис. 4.5: Сохранение ключа

Далее необходимо загрузить сгенерённый открытый ключ. Для этого захожу на сайт <http://github.org/> под своей учётной записью и перехожу в меню Setting . После этого выбираю в боковом меню SSH and GPG keys и нажимаю кнопку New SSH key Копирую из локальной консоли ключ в буфер обмена (рис. 4.6):

```

diserebryakova@fedora: ~$ ls ~/.ssh/
id_ed25519 id_ed25519.pub
diserebryakova@fedora: ~$ cat ~/.ssh/id_ed25519.pub | xclip -sel clip
diserebryakova@fedora: ~$ cat ~/.ssh/id_ed25519 | xclip -sel clip

```

Рис. 4.6: Копия ключа в буфер обмена

Вставляю скопированный ключ в поле «Key». В поле Title указываю имя для ключа. Нажимаю «Add SSH-key», чтобы завершить добавление ключа (рис. 4.7):

Add new SSH Key

Title

dserebryakova1902

Key type

Authentication Key

Key

ssh-ed25519 AAAAC3NzaC1lZD11NTE5AAAAIAA4ea0w5p1wXwAQ9w5NINglnVG/IB2ZGdfJfB83EA Dasha Serebryakova
<1132246733@pfur.ru>
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZkdjEAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAAAAtzc2gtZW
QyNTUxOQAAAC7wGuHmtMOadcf8AEPCOTYjVjZ1Rv4gdmRnRyXQINxAAAAALCVWHUJVoR
1AAAAAtzc2gtZWQyNTUxOQAAAC7wGuHmtMOadcf8AEPCOTYjVjZ1Rv4gdmRnRyXQINxAA
AAAEEDPFCultlZKxE0y4r62486T3bLo4qIRYcQGcZ7H7a5gubvAa4ea0w5p1wXwAQ9w5NIN
glnVG/IB2ZGdfJfB83EAAAAAJOHc2hhIFNlcmViclha292YS8MTExMj0NjczM0BwZn
VylN1PgECwQFBg==

Add SSH key

Рис. 4.7: Создание ключа

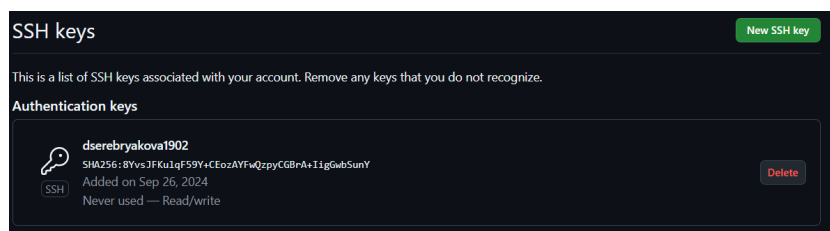


Рис. 4.8: Ключ создан

4.4 Создание рабочего пространства и репозитория курса на основе шаблона

Открываю терминал и создаю каталог для предмета «Операционные системы», используя команду `mkdir` с ключом `-p` (рис. 4.9):

```
dserebryakova@fedora: ~$ mkdir -p ~/work/study/2024-2025/"Операционные системы"
```

Рис. 4.9: Создание каталога для предмета

4.5 Создание репозитория курса на основе шаблона

Перехожу на страницу репозитория с шаблоном курса. Далее выбираю `Use this template`. В открывшемся окне задаю имя репозитория (Repository name) `study_2024-2025_os-intro` и создаю репозиторий. Репозиторий создан (рис. 4.10):

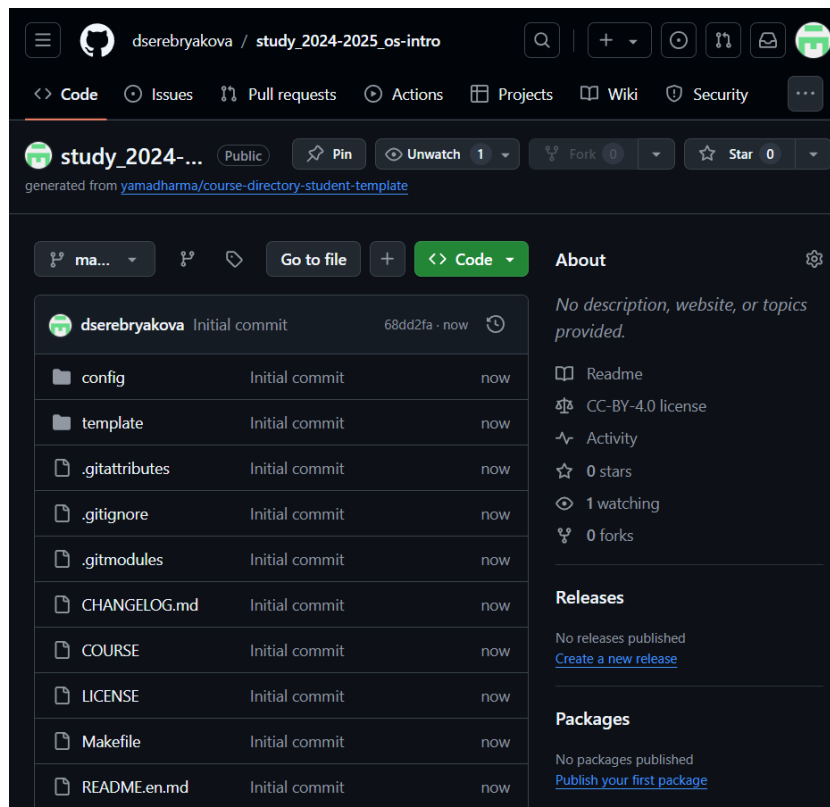


Рис. 4.10: Создание репозитория

Открываю терминал и перехожу в каталог курса (рис. 4.11):

```
dserebryakova@fedora: ~$ cd ~/work/study/2024-2025/"Операционные системы"
dserebryakova@fedora: ~/work/study/2024-2025/Операционные системы$
```

Рис. 4.11: Переход в каталог курса

Ссылку для клонирования копирую на странице созданного репозитория (рис. 4.12):

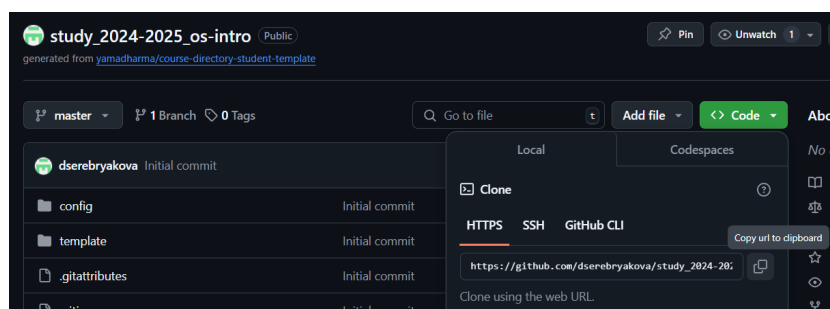


Рис. 4.12: Ссылка для клонирования

Клонирую созданный репозиторий, используя только что скопированную ссылку (рис. 4.13):

```
diserebryakova@fedora: ~/work/study/2024-2025/Операционные системы$ git clone --recursive  
git@github.com:dserebryakova/study_2024-2025_os-intro.git os-intro  
fatal: целевой путь «os-intro» уже существует и не является пустым каталогом.
```

Рис. 4.13: Клонирования созданного репозитория

4.6 Настройка каталога курса

Перехожу в каталог курса командой `cd` (рис. 4.14):

```
diserebryakova@fedora: ~$ cd ~/work/study/2024-2025/"Операционные системы"/os-intro  
diserebryakova@fedora: ~/work/study/2024-2025/Операционные системы/os-intro$
```

Рис. 4.14: Переход в каталог курса

Пробую удалить лишние файлы командой `rm` (рис. 4.15):

```
diserebryakova@fedora: ~/work/study/2024-2025/Операционные системы/os-intro$ rm package.js  
on  
rm: невозможно удалить 'package.json': Нет такого файла или каталога
```

Рис. 4.15: Удаление лишних файлов

Для создания необходимых каталогов использую команду `make` (рис. 4.16):

```
diserebryakova@fedora: ~/work/study/2024-2025/Операционные системы/os-intro$ make  
make: *** Не заданы цели и не найден make-файл. Останов.  
diserebryakova@fedora: ~/work/study/2024-2025/Операционные системы/os-intro$
```

Рис. 4.16: Создание необходимых каталогов

5 Вывод

В ходе выполнения работы изучены идеология и применение средств контроля версий. Приобретены практические навыки по работе с системой git

Список литературы

1. Архитектура ЭВМ