

Отчёт по лабораторной работе 2

Работа с github

Татьяна Соколова НММбд-03-24

Содержание

1	Цель работы	5
2	Задания	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
4.1	Знакомство с Markdown	9
5	Выводы	15
	Список литературы	16

Список иллюстраций

4.1	Шаблонный репозиторий	9
4.2	Создание репозитория	10
4.3	Мой репозиторий	10
4.4	Параметры git	11
4.5	Генерация ключа	11
4.6	Добавляю ключ в аккаунт	12
4.7	Добавляю ключ в аккаунт	12
4.8	Добавляю ключ в аккаунт	13
4.9	Создание папок курса	13
4.10	Загрузка	13
4.11	Загрузка	14

Список таблиц

1 Цель работы

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

2 Задания

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 Знакомство с Markdown

Регистрирую учетную запись на GitHub Приступаю к созданию репозитория на основе шаблона. (рис. 4.1, 4.2, 4.3)

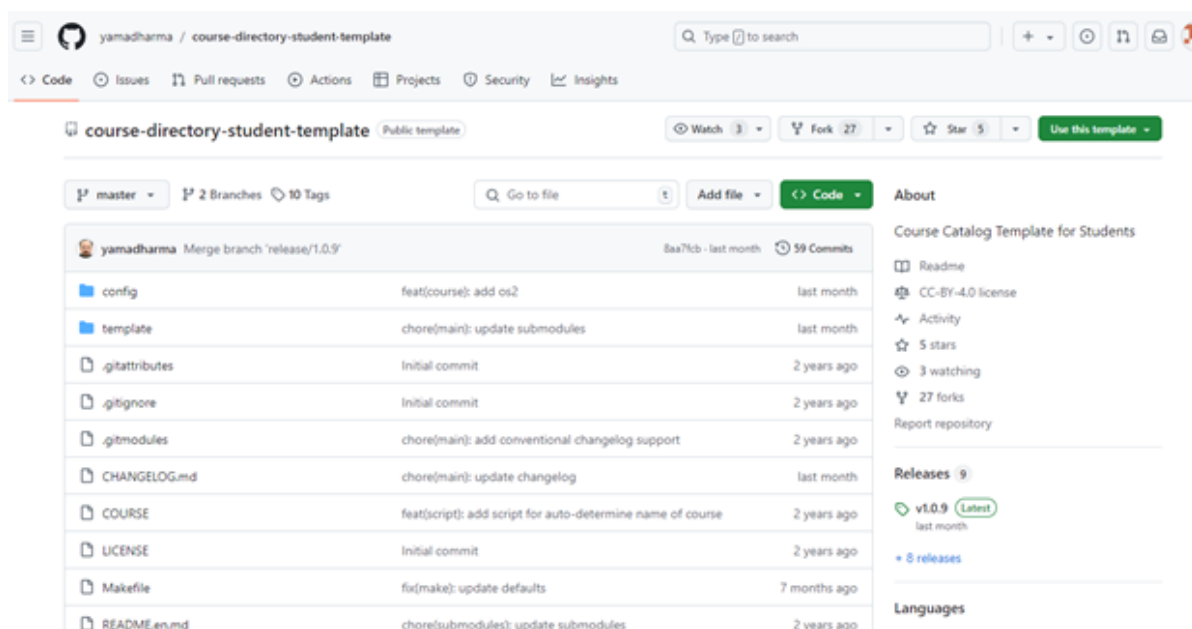



Рис. 4.1: Шаблонный репозиторий


Repository template

 **yamadharma/course-directory-student-template** ▾

Start your repository with a template repository's contents.

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


Owner * **Repository name ***


 **tssokolova** ▾ /


✔ arch-pc is available.

Great repository names are short and memorable. Need inspiration? How about **psychic-palm-tree** ?

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.




 You are creating a public repository in your personal account.


[Create repository](#)








Рис. 4.2: Создание репозитория

 **arch-pc** Public generated from [yamadharm/course-directory-student-template](#)


[Pin](#) [Unwatch 1](#) [Fork 0](#) [Star 0](#)


 master ▾  1 Branch  0 Tags


[Add file](#) [Code](#) [About](#) 


File	Commit	Time
 config	Initial commit	now
 template	Initial commit	now
 .gitattributes	Initial commit	now
 .gitignore	Initial commit	now
 .gitmodules	Initial commit	now
 CHANGELOG.md	Initial commit	now
 COURSE	Initial commit	now


No description, website, or topics provided.


 [Readme](#)

 [CC-BY-4.0 license](#)

 [Activity](#)

 [0 stars](#)

 [1 watching](#)

 [0 forks](#)

Releases

No releases published

[Create a new release](#)

Рис. 4.3: Мой репозиторий

Теперь подключимся к репозиторию из системы линукс. Для этого задаем параметры. (рис. 4.4)

```

tssokolova@Ubuntu:~$
tssokolova@Ubuntu:~$ git config --global user.name "tssokolova"

tssokolova@Ubuntu:~$ git config --global user.email "1132246764@pfur.ru"
tssokolova@Ubuntu:~$ git config --global core.quotepath false
tssokolova@Ubuntu:~$ git config --global init.defaultBranch master
tssokolova@Ubuntu:~$ git config --global core.autocrlf input
tssokolova@Ubuntu:~$ git config --global core.safecrlf warn
tssokolova@Ubuntu:~$

```

Рис. 4.4: Параметры git

SSH ключ нужен для авторизации пользователя. Создаем его (рис. 4.5)

```

tssokolova@Ubuntu:~$ ssh-keygen -C "tssokolova 1032245449@pfur.ru"
Generating public/private rsa key pair.

Enter file in which to save the key (/home/tssokolova/.ssh/id_rsa): Created directory '/home/tssokolova/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/tssokolova/.ssh/id_rsa
Your public key has been saved in /home/tssokolova/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:3/GE0DjkZ72SA/rgTtpfnKoz1Dge7ADdxHifYpVqkqQ tssokolova 1032245449@pfur.ru
The key's randomart image is:
+---[RSA 3072]-----+
|
|   o .
|  . o =
| + + =..
|E + *oo. o .
| . = +S = + .
| . *..B = =
|  =+o  O o .
|   ==O o o
|   ..==+
+-----[SHA256]-----+

```

Рис. 4.5: Генерация ключа

Теперь данные ключа нужно добавить в профиль на гитхабе. Тогда гитхаб будет узнавать нас по ключу. (рис. 4.6, 4.7)

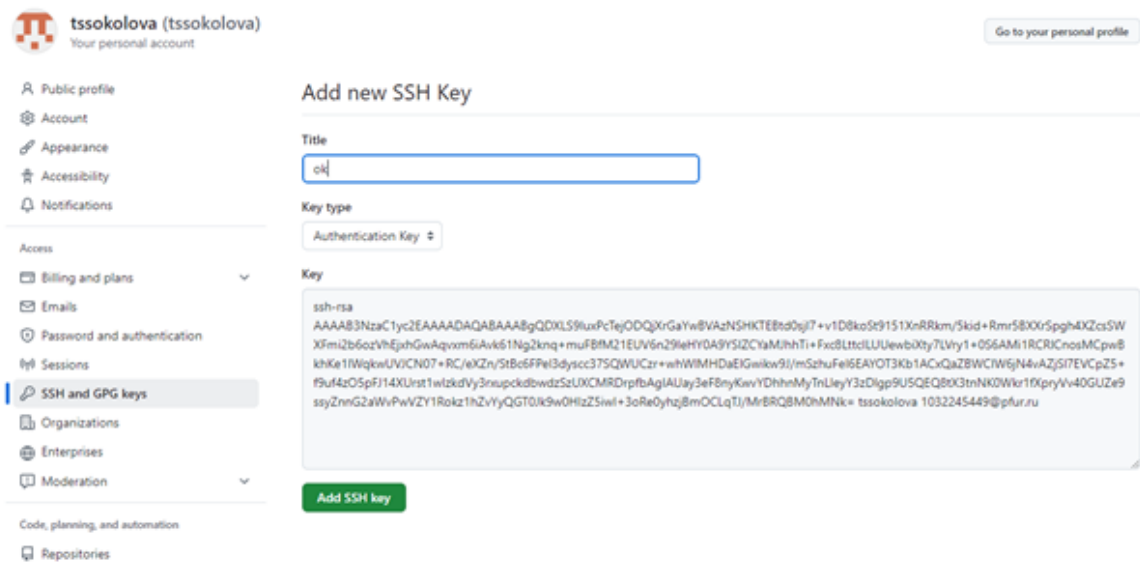


Рис. 4.6: Добавляю ключ в аккаунт

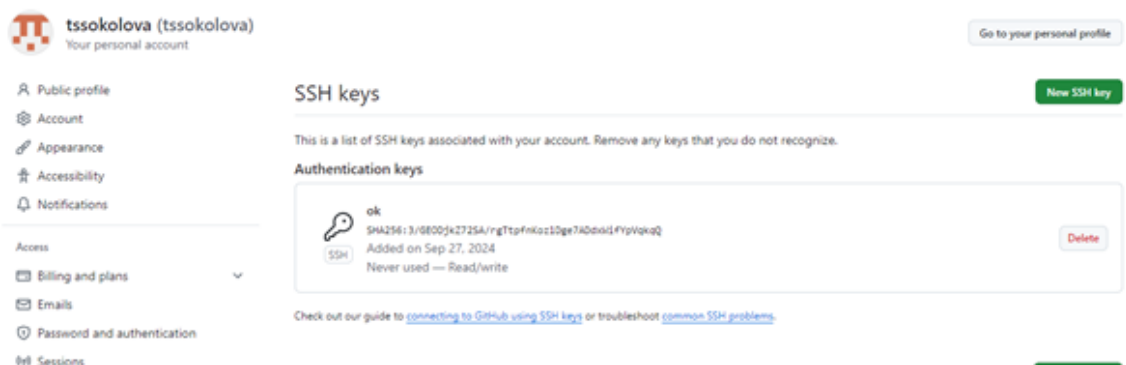


Рис. 4.7: Добавляю ключ в аккаунт

Создаем папку на компьютере и клонируем в нее содержимое репозитория, т е шаблон.(рис. 4.8)]

```

tssokolova@Ubuntu:~$
tssokolova@Ubuntu:~$ mkdir -p ~/work/study/2024-2025/"Архитектура компьютера"
tssokolova@Ubuntu:~$
tssokolova@Ubuntu:~$ cd ~/work/study/2024-2025/"Архитектура компьютера"
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера$
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера$ git clone --recursive git@github.
com:tssokolova/arch-pc.git
Cloning into 'arch-pc'...

The authenticity of host 'github.com (140.82.121.4)' can't be established.
ECDSA key fingerprint is SHA256:p2QAMXNIC1TJYWeI0ttrVc98/R1BUFWu3/LiyKgUfQM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com,140.82.121.4' (ECDSA) to the list of known hosts.
remote: Enumerating objects: 33, done.
remote: Counting objects: 100% (33/33), done.
remote: Compressing objects: 100% (32/32), done.
remote: Total 33 (delta 1), reused 18 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (33/33), 18.81 KiB | 6.27 MiB/s, done.
Resolving deltas: 100% (1/1), done.
Submodule 'template/presentation' (https://github.com/yamadharma/academic-presentation-markdown-tem
plate.git) registered for path 'template/presentation'
Submodule 'template/report' (https://github.com/yamadharma/academic-laboratory-report-template.git
) registered for path 'template/report'
Cloning into 'template/presentation'...
Cloning into 'template/report'...

```

Рис. 4.8: Добавляю ключ в аккаунт

Оформили курс по шаблону и загрузили в сетевой репозиторий (рис. 4.9, 4.10)

```

tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера$
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера$ cd ~/work/study/2024-2025/"Архите
ктура компьютера"/arch-pc
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$ rm package.json
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$ echo arch-pc > COURSE
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$ make prepare
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$ ls
CHANGELOG.md  COURSE  LICENSE  prepare  README.en.md  README.md
config        labs    Makefile  presentation  README.git-flow.md  template
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$

```

Рис. 4.9: Создание папок курса

```

create mode 100755 presentation/report/pandoc/filters/pandoc_secnos.py
create mode 100755 presentation/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 presentation/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 presentation/report/pandoc/filters/pandocxnos/core.py
create mode 100644 presentation/report/pandoc/filters/pandocxnos/main.py
create mode 100644 presentation/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 presentation/report/report.md
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$ git push
Enumerating objects: 37, done.
Counting objects: 100% (37/37), done.
Delta compression using up to 6 threads
Compressing objects: 100% (29/29), done.
Writing objects: 100% (35/35), 341.27 KiB | 2.46 MiB/s, done.
Total 35 (delta 4), reused 0 (delta 0)
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:tssokolova/arch-pc.git
 fcd6bb8..dff1c44 master -> master
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$

```

Рис. 4.10: Загрузка

Также загрузили в сетевой репозиторий отчеты по сделанным работам (рис. 4.11)

```
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$  
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$ git add .  
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$ git commit -am 'upload lab01'  
[master 0b5430c] upload lab01  
 1 file changed, 0 insertions(+), 0 deletions(-)  
 create mode 100644 labs/lab01/report/НММбд-03-24_Соколова_отчёт.pdf  
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$ git push  
Enumerating objects: 10, done.  
Counting objects: 100% (10/10), done.  
Delta compression using up to 6 threads  
Compressing objects: 100% (6/6), done.  
Writing objects: 100% (6/6), 779.52 KiB | 4.58 MiB/s, done.  
Total 6 (delta 3), reused 0 (delta 0)  
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.  
To github.com:tssokolova/arch-pc.git  
 dff1c44..0b5430c master -> master  
tssokolova@Ubuntu:~/work/study/2024-2025/Архитектура компьютера/arch-pc$
```

Рис. 4.11: Загрузка

5 Выводы

При выполнении данной лабораторной работы я изучила идеологию и применение средств контроля версий, а также приобрела практические навыки по работе с системой git.

Список литературы

1. Архитектура ЭВМ
2. Git - gitattributes Документация