

# **Лабораторная работа №2**

**Архитектура вычислительных систем**

Заболотная Кристина Александровна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>9</b>
<b>5</b>	<b>Выводы</b>	<b>16</b>
	<b>Список литературы</b>	<b>17</b>

## Список иллюстраций

4.1	рис.21 . . . . .	9
4.2	рис.22а . . . . .	9
4.3	рис.22б . . . . .	10
4.4	рис.23 . . . . .	10
4.5	рис.24а . . . . .	11
4.6	рис.24б . . . . .	11
4.7	рис.25 . . . . .	12
4.8	рис.26 . . . . .	12
4.9	рис.27а . . . . .	13
4.10	рис.27б . . . . .	13
4.11	рис.28 . . . . .	14
4.12	рис.29а . . . . .	14
4.13	рис.29б . . . . .	14
4.14	рис.29в . . . . .	14
4.15	рис.210 . . . . .	15

## Список таблиц

# 1 Цель работы

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

## 2 Задание

Создать отчет по выполнению лабораторной работы в соответствующем каталоге рабочего пространства (labs>lab03>report). Скопировать отчеты по выполнению предыдущих лабораторных работ в соответствующие каталоги созданного рабочего пространства. Загрузить файлы на github.

### 3 Теоретическое введение

- 1) Системы контроля версий. Общие понятия. Системы контроля версий 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 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.
- 2) Система контроля версий Git. Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями. Бла-

годаря тому, что Git является распределённой системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией.

- 3) Стандартные процедуры работы при наличии центрального репозитория. 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): `git checkout master` `git pull` `git checkout -b имя_ветки` Затем можно вносить изменения в локальном дереве и/или ветке. После завершения внесения какого-то изменения в файлы и/или каталоги проекта необходимо разместить их в центральном репозитории. Для этого необходимо проверить, какие файлы изменились к текущему моменту: `git status` и при необходимости удаляем лишние файлы, которые не хотим отправлять в центральный репозиторий. Затем полезно просмотреть текст изменений на предмет соответствия правилам ведения чистых коммитов: `git diff`. Если какие-либо файлы не должны попасть в коммит, то помечаем только те файлы, изменения которых нужно сохранить. Для этого используем команды добавления и/или удаления с нужными опциями: `git add имена_файлов` `git rm имена_файлов` Если нужно сохранить все изменения в текущем каталоге, то используем: `git add .` Затем сохраняем изменения, поясняя, что было сделано: `git commit -am "Some commit message"` и отправляем в центральный репозиторий: `git push origin имя_ветки` или `git push`.



## 4 Выполнение лабораторной работы

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.1)

- 1) Сначала сделаем предварительную конфигурацию git. Откроем терминал и введём команды, указав своё имя и email (как владельца репозитория).

```
kazaboltnaya@dk2n24 ~ $ git config --global user.name "<ChristinaZaboltnaya03>"
kazaboltnaya@dk2n24 ~ $ git config --global user.email "<christinazaboltnaya@mail.ru>"
```

Рис. 4.1: рис.21

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.2)

- 2) Настроим utf-8 в выводе сообщений git, зададим имя начальной ветки (будем называть её master), параметр autocrlf, параметр safecrlf.

```
kazaboltnaya@dk2n24 ~ $ git config --global core.quotePath false
kazaboltnaya@dk2n24 ~ $ git config --global init.defaultBranch master
```

Рис. 4.2: рис.22a

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.3)

Пункт 2 - продолжение.

```
kazabolotnaya@dk2n24 ~ $ git config --global core.autocrlf input
kazabolotnaya@dk2n24 ~ $ git config --global core.safecrlf warn
```

Рис. 4.3: рис.226

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.4)

- 3) Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый).

```
kazabolotnaya@dk2n24 ~ $ ssh-keygen -C "Christina Zabolotnaya03 <christinazabolotnaya@mail.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/afs/.dk.sci.pfu.edu.ru/home/k/a/kazabolotnaya/.ssh/id_rsa):
/afs/.dk.sci.pfu.edu.ru/home/k/a/kazabolotnaya/.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 /afs/.dk.sci.pfu.edu.ru/home/k/a/kazabolotnaya/.ssh/id_rsa
Your public key has been saved in /afs/.dk.sci.pfu.edu.ru/home/k/a/kazabolotnaya/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:ZQYu1HLkbjQLx2HfFHM9KXv9paUfW/CC2P+mImFrzHc Christina Zabolotnaya03 <christinazabolotnaya@mail.ru>
```

Рис. 4.4: рис.23

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.5)

- 4) Далее загрузим сгенерённый открытый ключ на сайте <http://github.org/> под своей учётной записью и перейдем в меню Setting . После этого выберем в боковом меню SSH and GPG keys и нажмем кнопку New SSH key. Скопируем из локальной консоли ключ в буфер обмена. Вставляем ключ в появившееся на сайте поле и указываем для ключа имя (Title).

## SSH keys / Add new

Title

Key type

Authentication Key ▾

Key

```
AAAAB3NzaC1yc2EAAAADAQABAAQGCnt1buCu6aDDe2U2FmZKNd1fN1wwJ7tWRCIIzkSgo5Cx28Oxtg7G
aU98yJ2RNebJlc/0wAgpEZ5yBHwVzGBfr+xY8jaQ8ZbPjK7TFk3Ta9wTFyXz1TZim1Q2UJlTpwka3VmewsOt78RI
EccHDWljvAtY53zXC6x/oZoXnKwfn8K1MigEpFD3nBhxGTQSADzn0U5bp1HESVJJEVId0fWq/GNvASB
/cajPiIWIq1vJrkio5r0EIHQHQ1OXPahghJgDB6eWQ2y2+4s1+Tcicbu2GUckQeUAnDhh1jQVpRVzjUT3He1qSNK
+EwQa5hseRv5CwADnlmF72lclTAG45R3GiyjOY5jyD8GIPTZQ5r+ccl
/HAQiCukOmIesmCdBE3VPAi9kWRHY1df4qwEEg5lQnZ1GXuXsqfZy1Pfa5UdoBnhLsFLnVluok3fQ0MeLF778p
vHqun3agWDChqHd+8qFwQ++GrC5GoghC3zTnRL7EnxnT82exttbY7A9EFwlzyec= Christina Zabolotnaya
<christi20222003@mail.ru>
```

Add SSH key

Рис. 4.5: рис.24а

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.6)

Пункт 4 - продолжение.

```
kazabolotnaya@dk2n24 ~ $ cat ~/.ssh/id_rsa.pub | xclip -sel clip
```

Рис. 4.6: рис.24б

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.7)

5) Создадим репозиторий, дадим ему название (study\_2022–2023\_arh-pc).


## Create a new repository from course-directory-student-template


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

Owner \* kazabolotnaya ▾ / Repository name \* study\_2022-2023\_arh-pc ✓


Great repository names are short and simple. Your new repository will be created as `study_2022-2023_arh-pc`. [j-funicular?](#)

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 `yamadharma/course-directory-student-template` and not just master.

 You are creating a public repository in your personal account.

[Create repository from template](#)

Рис. 4.7: рис.25

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.8)

- 6) Откроем терминал и создадим каталог для предмета «Архитектура компьютера», перейдём в каталог курса.

```
kazabolotnaya@dk2n26 ~ $ mkdir -p ~/work/study/2022-2023/"Архитектура компьютера"  
kazabolotnaya@dk2n26 ~ $ cd ~/work/study/2022-2023/"Архитектура компьютера"
```

Рис. 4.8: рис.26

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.9)

- 7) Клонировем созданный репозиторий (Ссылку для клонирования скопируем на странице созданного репозитория).

```
kazabolotnaya@dk2n24 ~/work/study/2022-2023/Архитектура компьютера $ git clone --recursive git@github.com:ChristinaZabolotnaya03/study_2022-2023_arch-pc.git arch-pc
Клонирование в «arch-pc»...
remote: Enumerating objects: 26, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (25/25), done.
remote: Total 26 (delta 0), reused 17 (delta 0), pack-reused 0
Получение объектов: 100% (26/26), 16.40 КиБ | 16.40 МБ/с, готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистрирован по пути «template/report»
Клонирование в «/afs/.dk.sci.pfu.edu.ru/home/k/a/kazabolotnaya/work/study/2022-2023/Архитектура компьютера/arch-pc/template/presentation»...
remote: Enumerating objects: 71, done.
remote: Counting objects: 100% (71/71), done.
remote: Compressing objects: 100% (49/49), done.
remote: Total 71 (delta 23), reused 68 (delta 20), pack-reused 0
Получение объектов: 100% (71/71), 88.89 КиБ | 1.06 МБ/с, готово.
Определение изменений: 100% (23/23), готово.
Клонирование в «/afs/.dk.sci.pfu.edu.ru/home/k/a/kazabolotnaya/work/study/2022-2023/Архитектура компьютера/arch-pc/template/report»...
remote: Enumerating objects: 78, done.
```

Рис. 4.9: рис.27а

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.10)

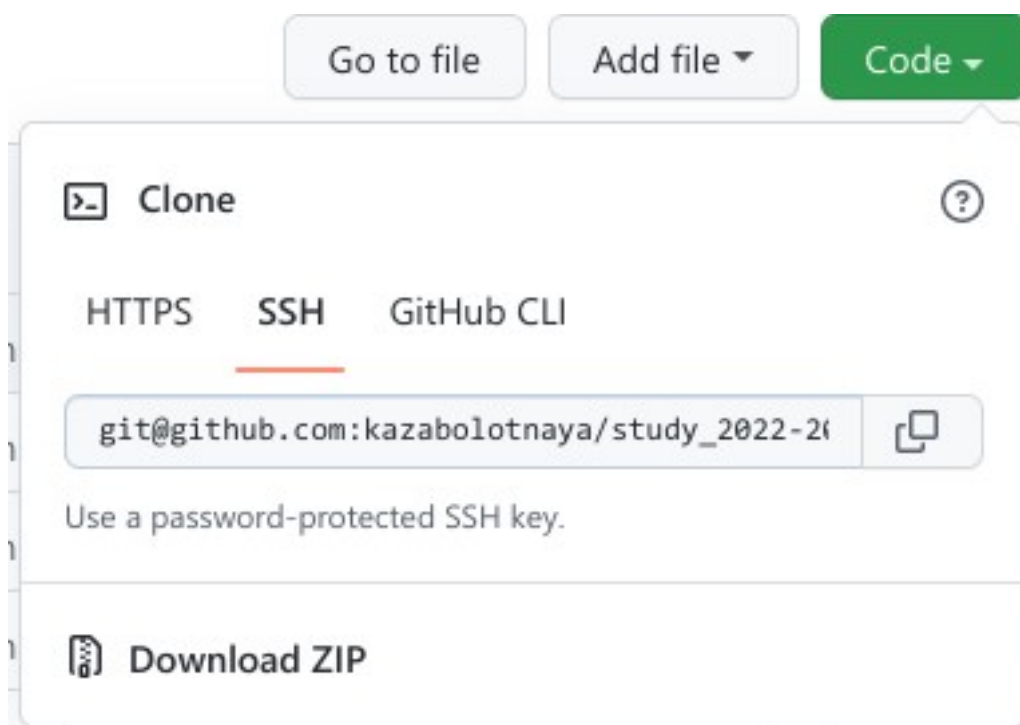


Рис. 4.10: рис.27б

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.11)

8) Перейдём в каталог курса. Удалим лишние файлы. Создадим необходимые

каталоги.

```
kazabolotnaya@dk2n24 ~/work/study/2022-2023/Архитектура компьютера $ cd ~/work/study/2022-2023/"Архитектура компью  
тера"/arch-pc  
kazabolotnaya@dk2n24 ~/work/study/2022-2023/Архитектура компьютера/arch-pc $ rm package.json  
kazabolotnaya@dk2n24 ~/work/study/2022-2023/Архитектура компьютера/arch-pc $ echo arch-pc > COURSE  
kazabolotnaya@dk2n24 ~/work/study/2022-2023/Архитектура компьютера/arch-pc $ make
```

Рис. 4.11: рис.28

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.12)

9) Отправим файлы на сервер.

```
kazabolotnaya@dk2n24 ~/work/study/2022-2023/Архитектура компьютера/arch-pc $ git add .  
kazabolotnaya@dk2n24 ~/work/study/2022-2023/Архитектура компьютера/arch-pc $
```

Рис. 4.12: рис.29а

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.13)

```
kazabolotnaya@dk2n24 ~/work/study/2022-2023/Архитектура компьютера/arch-pc $ git commit -am 'feat(main): make course structure'  
[master 3da7844] feat(main): make course structure  
91 files changed, 8229 insertions(+), 14 deletions(-)  
create mode 100644 labs/lab01/presentation/Makefile  
create mode 100644 labs/lab01/presentation/image/kulyabov.jpg  
create mode 100644 labs/lab01/presentation/presentation.md  
create mode 100644 labs/lab01/report/Makefile
```

Рис. 4.13: рис.29б

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.14)

```
kazabolotnaya@dk2n24 ~/work/study/2022-2023/Архитектура компьютера/arch-pc $ git push  
Перечисление объектов: 22, готово.  
Подсчет объектов: 100% (22/22), готово.  
При сжатии изменений используется до 6 потоков  
Сжатие объектов: 100% (16/16), готово.  
Запись объектов: 100% (20/20), 310.96 КиБ | 11.96 МиБ/с, готово.  
Всего 20 (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:ChristinaZabolotnaya03/study_2022-2023_arh-pc.git  
b392c2e..3da7844 master -> master
```

Рис. 4.14: рис.29в

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.15)

10) Отправим файлы на сервер.

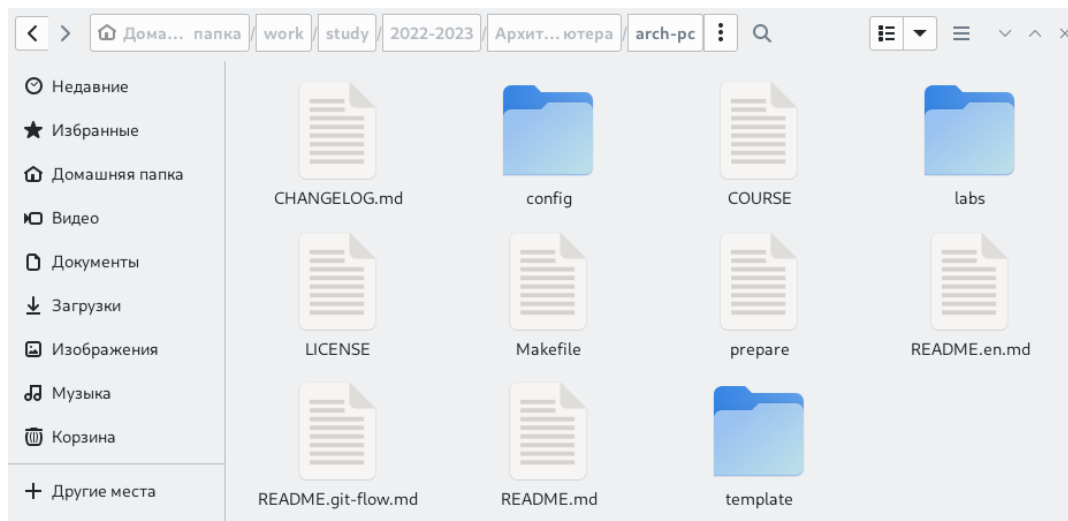


Рис. 4.15: рис.210

## 5 Выводы

В ходе изучения данной лабораторной работы были приобретены практические навыки по работе с системой git, научились создавать репозиторий, отправлять файлы на сервер. Изучили идеологию и применение средств контроля версий.



## **Список литературы**