

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

дисциплина: Архитектура компьютеров и операционных систем

Студент: Плугин Никита Сергеевич

Группа: НБИбд-02-23

МОСКВА

2023 г

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.....	14
4.5	Создание репозитория курса на основе шаблона	15
4.6	Настройка каталога курса.....	17
4.7	Выполнение заданий для самостоятельной работы	20
5	Выводы	27
6	Список литературы	28

1 Цель работы

Целью данной работы является изучить идеологию и применение средств контроля версий, а также 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 работ

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,

работающем 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 локальном дереве 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

Создаю учетную запись на сайте GitHub, далее я заполняю основные данные учетной записи и регистрирую аккаунт.

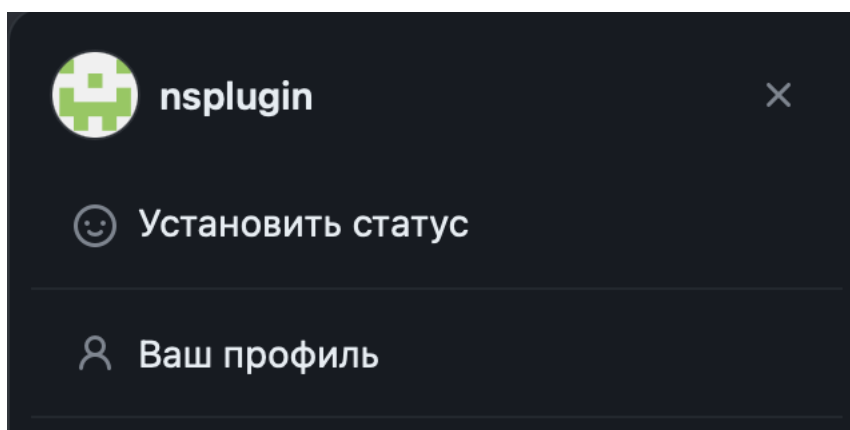


Рис. 4.2: Аккаунт GitHub

4.2 Базовая настройка Git

Запускаю виртуальную машину, затем в терминале задаю предварительную конфигурацию git. Ввожу команду `git config --global user.name ""`, указывая свое имя и команду `git config --global user.email "work@mail"`, указывая в ней электронную почту владельца, то есть мою

```
nsplugin@nsplugin:~$ git config --global user.name "<nsplugin>"
nsplugin@nsplugin:~$ git config --global user.email "<nikitaplugin@yandex.ru>"
nsplugin@nsplugin:~$
```

Рис. 4.3: Предварительная конфигурация git

Настраиваю utf-8 в выводе сообщений git для корректного отображения символов (рис. 4.4).

```
nsplugin@nsplugin:~$ git config --global core.quotePath false
nsplugin@nsplugin:~$ █
```

Рис. 4.4: Настройка кодировки

Задаю имя «master» для начальной ветки (рис.4.5).

```
nsplugin@nsplugin:~$ git config --global init.defaultBranch master
nsplugin@nsplugin:~$ █
```

Рис. 4.5: Создание имени для начальной ветки

Задаю параметр `autocrlf` со значением `input`, так как я работаю в системе Linux, чтобы конвертировать CRLF в LF только при коммитах (рис. 4.6). CR и LF – это символы, которые можно использовать для обозначения разрыва строки в текстовых файлах.


```
nsplugin@nsplugin:~$ git config --global core.autocrlf input
nsplugin@nsplugin:~$
```

Рис. 4.6: Параметр autocrlf

Задаю параметр safecrlf со значением warn, так Git будет проверять преобразование на обратимость (рис. 4.7). При значении warn Git только выведет предупреждение, но будет принимать необратимые конвертации.

```
nsplugin@nsplugin:~$ git config --global core.safecrlf warn
nsplugin@nsplugin:~$
```

Рис. 4.7: Параметр safecrlf

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 имя владельца и электронную почту владельца (рис. 4.8). Ключ автоматически сохранится в каталоге `~/.ssh/`.

```

nsplugin@nsplugin:~$ ssh-keygen -C "nsplugin <nikitaplugin@yandex.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/nsplugin/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/nsplugin/.ssh/id_rsa
Your public key has been saved in /home/nsplugin/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:wk/OLYWRL5db6ExWADDNqo09fPDTmyf6eCyJ/DnDY74 nsplugin <nikitaplugin@yande:
.ru>
The key's randomart image is:
+---[RSA 3072]-----+
|      +=+.....      |
|      . ++0. +       |
|      .o=0.* .       |
|      . o++*.o       |
|      o S...+o       |
|      * o  + .       |
|      . =0.+ o       |
|      o.Xoo         |
|      +E0.          |
+-----[SHA256]-----+
nsplugin@nsplugin:~$

```

Рис. 4.8: Генерация SSH-ключа

Xclip – утилита, позволяющая скопировать любой текст через терминал. Проверяю ее наличие с помощью команды apt-get install с ключом -у от имени суперпользователя, введя в начале команды sudo (рис. 4.9).

```

nsplugin@nsplugin:~$ sudo apt-get install -y xclip
[sudo] пароль для nsplugin:
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Следующие НОВЫЕ пакеты будут установлены:
  xclip
Обновлено 0 пакетов, установлено 1 новых пакетов, для удаления отмечено 0 пакетов, и 27 пакетов не обновлено.
Необходимо скачать 17,6 kB архивов.
После данной операции объём занятого дискового пространства возрастёт на 51,2 kB.
.
Пол:1 http://ports.ubuntu.com/ubuntu-ports jammy/universe arm64 xclip arm64 0.13-2 [17,6 kB]
Получено 17,6 kB за 0 с (72,2 kB/s)
Выбор ранее не выбранного пакета xclip.
(Чтение базы данных ... на данный момент установлено 169045 файлов и каталогов.)
Подготовка к распаковке .../xclip_0.13-2_arm64.deb ...
Распаковывается xclip (0.13-2) ...
Настраивается пакет xclip (0.13-2) ...
Обрабатываются триггеры для man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
nsplugin@nsplugin:~$ █

```

Рис. 4.9: Установка утилиты xclip

Копирую открытый ключ из директории, в которой он был сохранен, с помощью утилиты xclip (рис. 4.10).

```

nsplugin@nsplugin:~$ cat ~/lavtuxa.pub | xclip -sel clip
nsplugin@nsplugin:~$

```

Рис. 4.10: Копирование содержимого файла

Открываю браузер, захожу на сайт GitHub. Открываю свой профиль и выбираю страницу «SSH and GPG keys». Нажимаю кнопку «New SSH key»

Вставляю скопированный ключ в поле «Key». В поле Title указываю имя для ключа. Нажимаю «Add SSH-key», чтобы завершить добавление ключа (рис. 4.11).

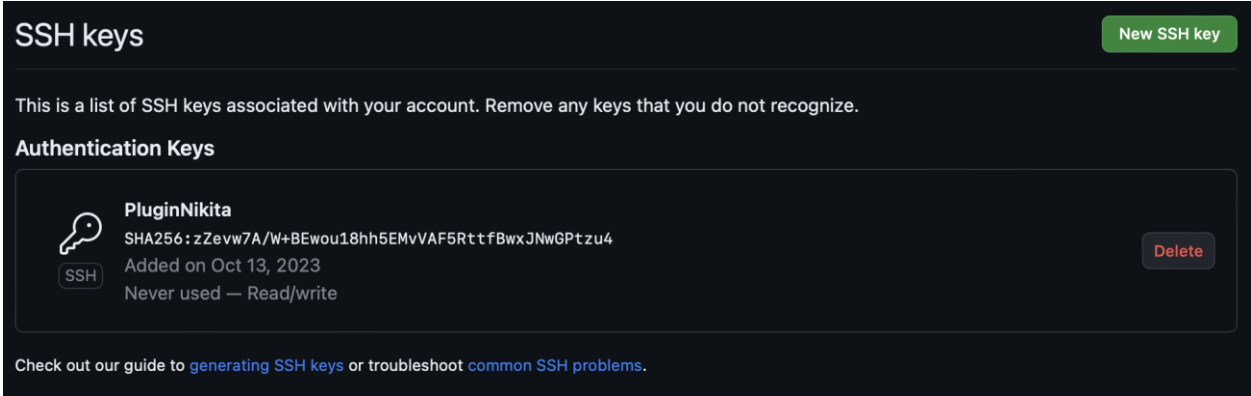


Рис. 4.11: Добавление ключа

4.4 Создание рабочего пространства и репозитория курса на основе шаблона

Закрываю браузер, открываю терминал. Создаю директорию, рабочее пространство, с помощью утилиты mkdir, благодаря ключу -p создаю все директории после домашней ~/work/study/2023-2024/“Computer architecture” рекурсивно. Далее проверяю с помощью ls, действительно ли были созданы необходимые мне каталоги (рис. 4.12).

```
nsplugin@nsplugin:~$ mkdir -p ~/work/study/2023-2024/"Computer architrcutre"
nsplugin@nsplugin:~$ ls
Documents      newdir         parentdir1     snap           Видео          Музыка          Шаблоны
lavruxa        parendir       parentdir2     tmp            Загрузки       Общедоступные
lavruxa.pub    parentdir      parentdir3     work           Изображения    'Рабочий стол'
```

Рис. 4.12: Создание рабочего пространства

Создание репозитория курса на основе шаблона

В браузере перехожу на страницу репозитория с шаблоном курса

по адресу <https://github.com/yamadharm/course-directory-student-template>. Далее выбираю «Use this template», чтобы использовать этот шаблон для своего репозитория (рис.4.13).

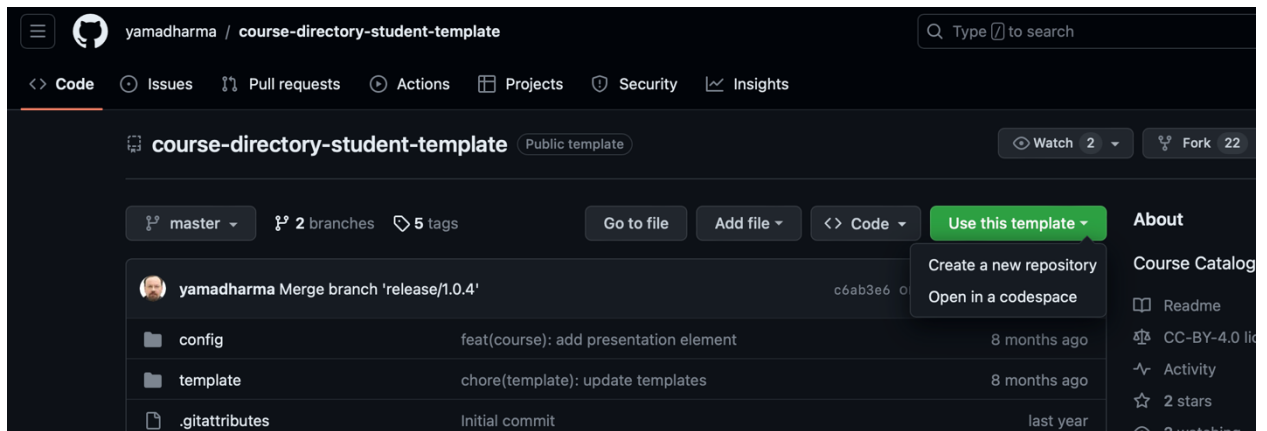


Рис. 4.13: Страница шаблона для репозитория


В открывшемся окне задаю имя репозитория (Repository name): study_2022–2023_arch-рс и создаю репозиторий, нажимаю на кнопку «Create repository» (рис. 4.14).

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Required fields are marked with an asterisk (*).


Owner * Repository name *


 nsplugin / lessons_2023-2024_ARCH


✔ lessons_2023-2024_ARCH is available.

Great repository names are short and memorable. Need inspiration? How about [bookish-octo-waffle](#) ?

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. 4.14: Окно создания репозитория

Через терминал перехожу в созданный каталог курса с помощью утилиты `cd`(рис. 4.15).

```
nsplugin@nsplugin:~$ cd ~/work/study/2023-2024/"Computer architrcure"  
nsplugin@nsplugin:~/work/study/2023-2024/Computer architrcure$
```

Рис. 4.15: Перемещение между директориями

Копирую ссылку для клонирования на странице созданного репозитория, сначала перейдя в окно «code», далее выбрав в окне вкладку «SSH»

Клонирую созданный репозиторий с помощью команды

git clone --recursive git@github.com:/study_2023–2024_arh-pc.git
arch-pc (рис. 4.16).

```
nsplugin@nsplugin:~/work/study/2023-2024/Computer architecture$ git clone --recursive git@github.com:nsplugin/lessons_2023-2024_ARCH.git arch-ps
Клонирование в «arch-ps»...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 27 (delta 1), reused 11 (delta 0), pack-reused 0
Получение объектов: 100% (27/27), 16.93 КиБ | 5.64 МиБ/с, готово.
Определение изменений: 100% (1/1), 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 «/home/nsplugin/work/study/2023-2024/Computer architecture/arch-ps/template/presentation»...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Получение объектов: 100% (82/82), 92.90 КиБ | 1022.00 КиБ/с, готово.
Определение изменений: 100% (28/28), готово.
Клонирование в «/home/nsplugin/work/study/2023-2024/Computer architecture/arch-ps/template/report»...
remote: Enumerating objects: 101, done.
remote: Counting objects: 100% (101/101), done.
remote: Compressing objects: 100% (70/70), done.
remote: Total 101 (delta 40), reused 88 (delta 27), pack-reused 0
Получение объектов: 100% (101/101), 327.25 КиБ | 1.69 МиБ/с, готово.
Определение изменений: 100% (40/40), готово.
Submodule path 'template/presentation': checked out 'b1be3800ee91f5809264cb755d316174540b753e'
Submodule path 'template/report': checked out '1d1b61dcac9c287a83917b82e3aef11a33b1e3b2'
```

Рис. 4.16: Клонирование репозитория

4.5 Настройка каталога курса

Удаляю лишние файлы с помощью утилиты rm (рис. 4.17).

```
nsplugin@nsplugin:~/work/study/2023-2024/Computer architecture/arch-ps$ rm package.json
nsplugin@nsplugin:~/work/study/2023-2024/Computer architecture/arch-ps$
```

Рис. 4.17: Удаление файлов

Создаю необходимые каталоги (рис. 4.18).

```
nsplugin@nsplugin:~/work/study/2023-2024/Computer architecture/arch-ps$ echo arch-pc > course
nsplugin@nsplugin:~/work/study/2023-2024/Computer architecture/arch-ps$ git add .
nsplugin@nsplugin:~/work/study/2023-2024/Computer architecture/arch-ps$ git commit -am 'feat(main): make course structure'
[master d5alceb] feat(main): make course structure
2 files changed, 1 insertion(+), 14 deletions(-)
create mode 100644 course
delete mode 100644 package.json
nsplugin@nsplugin:~/work/study/2023-2024/Computer architecture/arch-ps$ git push
Перечисление объектов: 4, готово.
Подсчет объектов: 100% (4/4), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (2/2), готово.
Запись объектов: 100% (3/3), 292 байта | 292.00 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:nsplugin/lessons_2023-2024_ARCH.git
5dd6b2c..d5alceb master -> master
nsplugin@nsplugin:~/work/study/2023-2024/Computer architecture/arch-ps$
```

Рис. 4.18: Создание каталогов

Самостоятельная работа:

Создадим отчёт по выполнению работы в каталоге рабочего пространства в lab01 и lab02. Загрузим файлы на github

Вывод:

Была изучена идеология и применение средств контроля версий, были приобретены практические навыки по работе с системой git, а также по работе сайте <https://github.com/>

Список литературы

1. Архитектура ЭВМ
2. Git - gitattributes Документация