

Отчёт по лабораторной работе №2

Дисциплина: Архитектура компьютеров

Лазарев Даниил Михайлович

Содержание

1	Цель работы	4
2	Задание	5
3	Теоретическое введение	6
4	Выполнение лабораторной работы	8
5	Создание SSH ключа	10
6	Создание рабочего пространства и репозитория курса на основе шаблона	12
7	Настройка каталога курса	14
8	Выполнение заданий для самостоятельной работы.	16
9	Выводы	18

Список иллюстраций

4.1	Имя и почта пользователя	8
4.2	Вывод utf-8	8
4.3	Начальная ветка «master»	8
4.4	Параметр “autocrlf”	9
4.5	Параметр “safecrlf”	9
5.1	Команда «ssh-keygen» и сам ключ	10
5.2	Команда “cat”	10
5.3	Ключ на Github	11
6.1	Команда “mkdir”	12
6.2	Команда “cd”	12
6.3	Результат команды «git clone»	13
7.1	Удаление файла «package.json»	14
7.2	Команды «echo» и «make»	14
7.3	Отправка файлов на сервер	15
8.1	lab02 на Github	16
8.2	lab01 на Github	17

1 Цель работы

Целью данной работы является изучение идеологии и применение средств контроля версий. Приобретение практических навыков 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. Задание 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 (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. 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. 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.

4 Выполнение лабораторной работы

Настройка GitHub

Поскольку учетная запись на Github уже имеется, регистрировать ее нет необходимости.

Базовая настройка git Откроем терминал и введем следующие команды с указанием имени и фамилии, а также нашей электронной почты. (рис. 4.1)

```
dmlazarev@dmlazarev-VirtualBox:~$ git config --global user.name "<Даниил Лазарев
>"
dmlazarev@dmlazarev-VirtualBox:~$ git config --global user.email "<anarchy1928@ma
il.ru>"
dmlazarev@dmlazarev-VirtualBox:~$
```

Рис. 4.1: Имя и почта пользователя

Настроим вывод utf-8 в выводе сообщений git. (рис. 4.2)

```
dmlazarev@dmlazarev-VirtualBox:~$ git config --global core.quotePath false
dmlazarev@dmlazarev-VirtualBox:~$
```

Рис. 4.2: Вывод utf-8

Так же зададим имя начальной ветки – «master». (рис. 4.3)

```
dmlazarev@dmlazarev-VirtualBox:~$ git config --global init.defaultBranch master
dmlazarev@dmlazarev-VirtualBox:~$
```

Рис. 4.3: Начальная ветка «master»

Так же подключим параметры «autocrlf» и «safecrlf». (рис. 4.4, рис. 4.5)


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

Рис. 4.4: Параметр “autocrlf”

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

Рис. 4.5: Параметр “safecrlf”

5 Создание SSH ключа

Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый). Для этого используем команду «ssh-keygen». (рис. 5.1). После генерации ключ сохраняется в каталоге ~/.ssh/.

```
dmlazarev@dmlazarev-VirtualBox:~$ ssh-keygen -C "Даниил Лазарев <anarchy1928@mail.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/dmlazarev/.ssh/id_rsa):
Created directory '/home/dmlazarev/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/dmlazarev/.ssh/id_rsa
Your public key has been saved in /home/dmlazarev/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:viWPK2689vfn0e9yUYtiIyKNCIJ0ufJej2bMfifcetw Даниил Лазарев <anarchy1928@mail.ru>
The key's randomart image is:
+---[RSA 3072]-----+
|
|o
|o.
|o o . o S . o|
|+ . o + . + . o|
| .o.O.o+. + o .|
|.. .o0 =*E +..|
|.oo+o==o=0...oo*o|
+---[SHA256]-----+
dmlazarev@dmlazarev-VirtualBox:~$
```

Рис. 5.1: Команда «ssh-keygen» и сам ключ

Скопируем получившийся ключ с помощью команды «cat» и загрузим его в наш аккаунт Github, указав имя для этого ключа.(рис. 5.2)(рис. 5.3)

```
dmlazarev@dmlazarev-VirtualBox:~$ cat ~/.ssh/id_rsa.pub | xclip -sel clip
dmlazarev@dmlazarev-VirtualBox:~$
```

Рис. 5.2: Команда “cat”

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



danechkavibe
SHA256:v1wPK2689vfn0e9yUYtiIyKNCIj0ufJej2bMfifcetw
Added on Sep 29, 2023
Never used — Read/write

Delete

Рис. 5.3: Ключ на Github

6 Создание рабочего пространства и репозитория курса на основе шаблона

При выполнении лабораторных работ следует соблюдать определенную

иерархию, которая у нас и соблюдается. Создадим каталог для предмета «Архитектура компьютера» с помощью терминала.(рис. 6.1)

```
dmlazarev@dmlazarev-VirtualBox:~$ mkdir -p ~/work/study/2023-2024/"Архитектура к  
омпьютера"  
dmlazarev@dmlazarev-VirtualBox:~$
```

Рис. 6.1: Команда “mkdir”

Создание репозитория курса на основе шаблона

Перейдя на страницу с шаблоном курса (<https://github.com/yamadharma/course-directory-student-template>)

создадим репозиторий и присвоим ему имя. Используя терминал перейдем к каталогу курса. (рис. 6.2)

```
dmlazarev@dmlazarev-VirtualBox:~$ cd ~/work/study/2023-2024/"Архитектура компьют  
ера"
```

Рис. 6.2: Команда “cd”

Клонируем данный репозиторий используя команду «git clone --recursive»

предварительно скопировав ссылку для клонирования в нашем личном кабинете за счет SSH-ключа.(рис. 6.3)

```
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 | 2.11 МБ/с, готово.
Определение изменений: 100% (40/40), готово.
Submodule path 'template/presentation': checked out 'b1be3800ee91f5809264cb755d3
16174540b753e'
Submodule path 'template/report': checked out '1d1b61dcac9c287a83917b82e3aef11a3
3b1e3b2'
dmlazarev@dmlazarev-VirtualBox:~/work/study/2023-2024/Архитектура компьютера$
```

Рис. 6.3: Результат команды «git clone»

7 Настройка каталога курса

Перейдем в каталог курса используя команду «cd» и удалим файл

«package.json» используя команду «rm». (рис. 7.1)

```
dmlazarev@dmlazarev-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arch-pc$ rm package.json
dmlazarev@dmlazarev-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arch-pc$
```

Рис. 7.1: Удаление файла «package.json»

Создаем необходимые нам каталоги в репозитории используя команду «echo»

и «make». (рис. 7.2).

```
dmlazarev@dmlazarev-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arch-pc$ echo arch-pc > COURSE
dmlazarev@dmlazarev-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arch-pc$ make
```

Рис. 7.2: Команды «echo» и «make»

После всех сделанных действий отправляем файлы на сервер используя

череду команд «git add», «git commit» и «git push». (рис. 7.3).

```
create mode 100644 presentation/report/report.md
dmlazarev@dmlazarev-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc$ git push
Перечисление объектов: 37, готово.
Подсчет объектов: 100% (37/37), готово.
При сжатии изменений используется до 8 потоков
Сжатие объектов: 100% (29/29), готово.
Запись объектов: 100% (35/35), 342.17 КиБ | 2.46 МиБ/с, готово.
Всего 35 (изменений 4), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:danechkavibe/study_2023-2024_arh-pc.git
   ecd7a62..41a5efe master -> master
dmlazarev@dmlazarev-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc$
```


Рис. 7.3: Отправка файлов на сервер

8 Выполнение заданий для самостоятельной работы.

Отчет по проведенной лабораторной работе создан и загружен в

(labs>lab02>report). (рис. 8.1). Так же предыдущая лабораторная работа была загружена в (labs>lab01>report). (рис. 8.2).

[study_2023-2024_arh-pc](#) / [labs](#) / [lab02](#) / **report** /

 **danechkavibe** feat(main): make course structure

Name	Last commit message
..	
bib	feat(main): make course structure
image	feat(main): make course structure
pandoc	feat(main): make course structure
Makefile	feat(main): make course structure
report.md	feat(main): make course structure
ЛР02_Лазарев_отчет.pdf	feat(main): make course structure

Рис. 8.1: lab02 на Github

[study_2023-2024_arh-pc](#) / [labs](#) / [lab01](#) / [report](#) / 









 danechkavibe feat(main): make course structure	
Name	Last commit message
 ..	
 bib	feat(main): make course structure
 image	feat(main): make course structure
 pandoc	feat(main): make course structure
 Makefile	feat(main): make course structure
 report.md	feat(main): make course structure
 ЛР01_Лазарев_отчет.pdf	feat(main): make course structure

Рис. 8.2: lab01 на Github

9 Выводы

В ходе лабораторной работы я изучила идеологию и применение средств контроля версий и приобрела практические навыки по работе с системой git.