

Отчёт по лабораторной работе №2

Первоначальная настройка гитхаб

Дагделен Зейнап Реджеповна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
3.1	Системы контроля версий. Общие понятия	7
3.2	Примеры использования git	8
3.3	Основные команды git	9
4	Выполнение лабораторной работы	10
4.1	Установка программного обеспечения	10
4.1.1	Установка git	10
4.2	Базовая настройка git	10
4.3	Создание ключей ssh и pgp	11
4.4	Настройка github	12
4.5	Добавление PGP ключа в GitHub	13
4.6	Настройка автоматических подписей коммитов git	14
4.7	Настройка gh	14
4.8	Создание репозитория курса на основе шаблона	15
4.9	Настройка каталога курса	15
5	Ответы на контрольные вопросы.	17
6	Выводы	19
	Список литературы	20

Список иллюстраций

4.1	Установка git	10
4.2	имя владельца репозитория	10
4.3	email владельца репозитория (первая строчка)	11
4.4	Настройка utf-8 (вторая строчка)	11
4.5	имя начальной ветки(третья строчка)	11
4.6	Настройка	11
4.7	Создание ключа	12
4.8	Создание ключа	12
4.9	Создание ключа	13
4.10	Копирование ключа	13
4.11	Сохранение GPG ключа	14
4.12	Подпись коммитов с использованием почты	14
4.13	Авторизация на сайте	15
4.14	Создание каталога	15
4.15	Создание курса на Git	15
4.16	Клонирование из шаблона в os-intro	15
4.17	Создание курса	16
4.18	Создание курса на Git	16

Список таблиц

1 Цель работы

- 1) Изучить идеологию и применение средств контроля версий.
- 2) Освоить умения по работе с git.

2 Задание

- 1) Установка программного обеспечения
- 2) Базовая настройка git
- 3) Создание ключей ssh и pgr
- 4) Настройка github
- 5) Добавление PGP ключа в GitHub
- 6) Настройка автоматических подписей коммитов git
- 7) Настройка gh
- 8) Шаблон для рабочего пространства

3 Теоретическое введение

3.1 Системы контроля версий. Общие понятия

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. 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. Кроме того, 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.

3.2 Примеры использования 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.
- Благодаря тому, что Git является распределённой системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией.

3.3 Основные команды git

Перечислим наиболее часто используемые команды git.

- Создание основного дерева репозитория: `git init`
- Получение обновлений (изменений) текущего дерева из центрального репозитория: `git pull`
- Отправка всех произведённых изменений локального дерева в центральный репозиторий: `git push`
- Просмотр списка изменённых файлов в текущей директории: `git status`

и тд.

4 Выполнение лабораторной работы

4.1 Установка программного обеспечения

4.1.1 Установка git

Устанавливаю git(рис. 4.1).

```
rzdagdelen@rzdagdelen:~$ git config --global user.name "Zeynap Dagdelen"
Команда «git» не найдена, но может быть установлена с помощью:
sudo apt install git
rzdagdelen@rzdagdelen:~$ sudo apt install git
[sudo] пароль для rzdagdelen:
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Будут установлены следующие дополнительные пакеты:
  git-man liberror-perl
Предлагаемые пакеты:
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb
  git-cvs git-mediawiki git-svn
Следующие НОВЫЕ пакеты будут установлены:
  git git-man liberror-perl
Обновлено 0 пакетов, установлено 3 новых пакетов, для удаления отмечено 0 пакетов,
и 259 пакетов не обновлено.
Необходимо скачать 4 147 kB архивов.
После данной операции объём занятого дискового пространства возрастёт на 21,0 MB
```

Рис. 4.1: Установка git

4.2 Базовая настройка git

Зададу имя и email владельца репозитория: с помощью команд “git config –global user.name”Zeynap Dagdelen”” и “git config –global user.email”zdagdelenn@gmail.com”” (рис. 4.2- 4.3).

```
rzdagdelen@rzdagdelen:~$ git config --global user.name "Zeynap Dagdelen"
```

Рис. 4.2: имя владельца репозитория

```

rzdagdelen@rzdagdelen:~$ git config --global user.email "zdagdelenn@gmail.com"
rzdagdelen@rzdagdelen:~$ git config --global core.quotepath false
rzdagdelen@rzdagdelen:~$ git config --global init.defaultBranch master
rzdagdelen@rzdagdelen:~$

```

Рис. 4.3: email владельца репозитория (первая строчка)

Настрою utf-8 в выводе сообщений git (рис. 4.4).

```

rzdagdelen@rzdagdelen:~$ git config --global user.email "zdagdelenn@gmail.com"
rzdagdelen@rzdagdelen:~$ git config --global core.quotepath false
rzdagdelen@rzdagdelen:~$ git config --global init.defaultBranch master
rzdagdelen@rzdagdelen:~$

```

Рис. 4.4: Настройка utf-8 (вторая строчка)

Зададу имя начальной ветки (будем называть её master)(рис. 4.5).

```

rzdagdelen@rzdagdelen:~$ git config --global user.email "zdagdelenn@gmail.com"
rzdagdelen@rzdagdelen:~$ git config --global core.quotepath false
rzdagdelen@rzdagdelen:~$ git config --global init.defaultBranch master
rzdagdelen@rzdagdelen:~$

```

Рис. 4.5: имя начальной ветки(третья строчка)

Настроим параметры autocrlf и safecrlf (рис. 4.6).

```

rzdagdelen@rzdagdelen:~$ git config --global core.autocrlf input
rzdagdelen@rzdagdelen:~$ git config --global core.safecrlf warn
rzdagdelen@rzdagdelen:~$

```

Рис. 4.6: Настройка

4.3 Создание ключей ssh и pgr

Создаю ключи ssh по алгоритму ed25519 (рис. 4.7).

```

rzdagdelen@rzdagdelen:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/rzdagdelen/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/rzdagdelen/.ssh/id_ed25519
Your public key has been saved in /home/rzdagdelen/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:HHxqbmYww1LeB+uj5In3l6bRYvufAk//PqMEUk27Nd8 rzdagdelen@rzdagdelen
The key's randomart image is:
+--[ED25519 256]--+
|
| . o .
| . + o o o
| + o B o o .
| . * S o . E
| . B + .
| . @ + . o .
| + . B = * . . o
| . . + . = . o + + o o
+-----[SHA256]-----+
rzdagdelen@rzdagdelen:~$ xclip -i < ~/.ssh/id_ed25519.pub

```

Рис. 4.7: Создание ключа

Создаю ключи pgr, для этого генерирую ключ с помощью 'gpg --full-generate-key' (рис. 4.8).

```

rzdagdelen@rzdagdelen:~$ gpg --full-generate-key
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Выберите тип ключа:
  (1) RSA и RSA (по умолчанию)
  (2) DSA и Elgamal
  (3) DSA (только для подписи)
  (4) RSA (только для подписи)
  (14) Имеющийся на карте ключ
Ваш выбор? 1
длина ключей RSA может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072) 4096
Запрошенный размер ключа - 4096 бит
Выберите срок действия ключа.
  0 = не ограничен
  <n> = срок действия ключа - n дней
  <n>w = срок действия ключа - n недель
  <n>m = срок действия ключа - n месяцев
  <n>y = срок действия ключа - n лет
Срок действия ключа? (0) 0
Срок действия ключа не ограничен
Все верно? (у/Н) у

```

Рис. 4.8: Создание ключа

4.4 Настройка github

Я создала учётную запись на <https://github.com> и заполнила основные данные на <https://github.com> (в прошлом курсе).

4.5 Добавление PGP ключа в GitHub

Вывожу список ключей и копирую отпечаток приватного ключа с помощью ‘gpg –list-secret-keys –keyid-format LONG’ (рис. 4.9).

```
rzdagdelen@rzdagdelen:~$ gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: глубина: 0 достоверных: 1 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f, 1
u
/home/rzdagdelen/.gnupg/pubring.kbx
-----
sec   rsa4096/0463ED9D400F8FAE 2024-02-25 [SC]
      0C4FED47D81C07E3BAF2540C0463ED9D400F8FAE
uid   [ абсолютно ] Zeynep Dagdelen <zdagdelen@gmail.com>
ssb   rsa4096/F5C6EA8D9DD187EB 2024-02-25 [E]

rzdagdelen@rzdagdelen:~$
```

Рис. 4.9: Создание ключа

Отпечаток ключа — это последовательность байтов, используемая для идентификации более длинного, по сравнению с самим отпечатком ключа.

Скопирую свой сгенерированный PGP ключ в буфер обмена с помощью ‘gpg –armor –export 0463ED9D400F8FAE | xclip -sel clip’(рис. 4.10).

```
rzdagdelen@rzdagdelen:~$ gpg --armor --export 0463ED9D400F8FAE | xclip -sel clip
```

Рис. 4.10: Копирование ключа


Перехожу в настройки GitHub (<https://github.com/settings/keys>), нажимаю на кнопку New GPG key и вставляю полученный ключ в поле ввода (рис. 4.11).

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

SSH


zrdagdelen2209
SHA256:paTzsdqTghn/1hrpcop9RCN2YjQ0hr2VB1wx+ZR0Dg8
Added on Sep 20, 2023
Last used within the last 3 months — Read/write

Delete

SSH

imaczey1110
SHA256:8vGLp6t+00mC/uqL1jQ/nxFidGcouNPgL0mVW9p3JYs
Added on Oct 11, 2023
Last used within the last 5 months — Read/write

Delete

SSH

zmacgit24
SHA256:oY18B4r5a0G61AT7yCuDFCcJbxIbzfBukrKDhmfPxT8
Added on Feb 25, 2024
Never used — Read/write


Delete

Check out our guide to [connecting to GitHub using SSH keys](#) or troubleshoot [common SSH problems](#).

GPG keys

New GPG key

This is a list of GPG keys associated with your account. Remove any keys that you do not recognize.

GPG

zeymac24
Email address: zdagdelen@gmail.com
Key ID: 0463ED9D400F8FAE
Subkeys: F5C6EA8D9DD187EB
Added on Feb 25, 2024

Delete

Learn how to [generate a GPG key and add it to your account](#).

Рис. 4.11: Сохранение GPG ключа

4.6 Настройка автоматических подписей коммитов git

Используя введённый email, указываю Git применять его при подписи коммитов (рис. 4.12).

```
rzdagdelen@rzdagdelen:~$ git config --global user.signingkey 0463ED9D400F8FAE
rzdagdelen@rzdagdelen:~$ git config --global commit.gpgsign true
rzdagdelen@rzdagdelen:~$ git config --global gpg.program $(which gpg2)
rzdagdelen@rzdagdelen:~$
```

Рис. 4.12: Подпись коммитов с использованием почты

4.7 Настройка gh

Для начала необходимо авторизоваться с помощью ‘gh auth login’ (рис. 4.13).

```
rzdagdelen@rzdagdelen:~$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? SSH
? Generate a new SSH key to add to your GitHub account? No
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 0D75-1F2C
Press Enter to open github.com in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
✓ Logged in as zrdagdelen
rzdagdelen@rzdagdelen:~$
```

Рис. 4.13: Авторизация на сайте

4.8 Создание репозитория курса на основе шаблона

Необходимо создать шаблон рабочего пространства с помощью последовательности команд: (рис. 4.14, 4.15, 4.16).

```
rzdagdelen@rzdagdelen:~$ mkdir -p ~/work/study/2022-2023/"Операционные системы"
rzdagdelen@rzdagdelen:~$ mkdir -p ~/work/study/2023-2024/"Операционные системы"
rzdagdelen@rzdagdelen:~$ cd ~/work/study/2023-2024/"Операционные системы"
```

Рис. 4.14: Создание каталога

```
tudy_2023-2024_os-intro --template=yamadharma/course-directory-student-template --p
ublic
✓ Created repository zrdagdelen/study_2023-2024_os-intro on GitHub
rzdagdelen@rzdagdelen:~/work/study/2023-2024/Операционные системы$ git clone --recu
```

Рис. 4.15: Создание курса на Git

```
rzdagdelen@rzdagdelen:~/work/study/2023-2024/Операционные системы$ git clone --recu
rsive git@github.com:zrdagdelen/study_2023-2024_os-intro.git os-intro
Клонирование в «os-intro»...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhbpZ1sF/zLDA0zPMSvHdkr4UVC0qU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
```

Рис. 4.16: Клонирование из шаблона в os-intro

4.9 Настройка каталога курса

Перехожу в каталог курса спомощью cd, удаляю лишние файлы с помощью rm и создаю необходимые каталоги (рис. 4.17).

```

rzdagdelen@rzdagdelen:~/work/study/2023-2024/Операционные системы$ cd os-intro
rzdagdelen@rzdagdelen:~/work/study/2023-2024/Операционные системы/os-intro$ rm package.json
rzdagdelen@rzdagdelen:~/work/study/2023-2024/Операционные системы/os-intro$ echo os-intro > COURSE
rzdagdelen@rzdagdelen:~/work/study/2023-2024/Операционные системы/os-intro$ make
Usage:
  make <target>

Targets:
  list           List of courses
  prepare       Generate directories structure
  submodule      Update submodules
rzdagdelen@rzdagdelen:~/work/study/2023-2024/Операционные системы/os-intro$

```

Рис. 4.17: Создание курса

Отправляю файлы на сервер (рис. 4.18).

```

rzdagdelen@rzdagdelen:~/work/study/2023-2024/Операционные системы/os-intro$ git add .
rzdagdelen@rzdagdelen:~/work/study/2023-2024/Операционные системы/os-intro$ git commit -am 'feat(main): make course structure'
[master bf86bea] feat(main): make course structure
2 files changed, 1 insertion(+), 14 deletions(-)
delete mode 100644 package.json
rzdagdelen@rzdagdelen:~/work/study/2023-2024/Операционные системы/os-intro$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 954 bytes | 954.00 KiB/s, done.
Total 3 (insertions 1), 0 (deletions 0), 0 (updates 0), 0 (packs 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:rzdagdelen/study_2023-2024_os-intro.git
a7bb9e9..bf86bea master -> master
rzdagdelen@rzdagdelen:~/work/study/2023-2024/Операционные системы/os-intro$

```

Рис. 4.18: Создание курса на Git

5 Ответы на контрольные вопросы.

- 1) Система контроля версий (VCS) - это программное обеспечение, которое позволяет отслеживать изменения в документах и проектах, при необходимости производить их откат, сохранять историю изменений, а также управлять параллельной разработкой кода несколькими участниками.
- 2) Хранилище (repository) - это специальное хранилище файлов и папок проекта, изменения в которых отслеживаются. Коммит (commit) - это процесс сохранения изменений в хранилище. История (history) - это полная хронология изменений в коде. Рабочая копия (working copy) - это местная копия кода, которая находится на компьютере разработчика перед тем, как она будет сохранена в хранилище.
- 3) Централизованные VCS используют один общедоступный сервер хранилища, через который производятся все изменения. Децентрализованные VCS позволяют каждой копии хранилища быть независимой и автономной, изменения могут быть синхронизированы между копиями хранилища, но не требуют участия центрального сервера. Примеры централизованных VCS: CVS, SVN. Примеры децентрализованных VCS: Git, Mercurial.
- 4) При единоличной работе с хранилищем действия с VCS включают клонирование репозитория на локальную машину, создание изменений в рабочей копии и коммит изменений в хранилище.
- 5) Порядок работы с общим хранилищем VCS включает клонирование репозитория на локальную машину, создание изменений в рабочей копии, коммит изменений в локальный репозиторий, отправку изменений в удаленный

репозиторий и слияние изменений с другими ветками.

- 6) Основные задачи, решаемые инструментальным средством Git, включают управление версиями, управление ветвлением, управление изменениями, управление конфликтами и управление совместной работой.
- 7) Некоторые команды Git и их краткая характеристика: `git init` - создание нового репозитория, `git add` - добавление изменений в индекс, `git commit` - сохранение изменений в репозитории, `git push` - отправка изменений в удаленный репозиторий, `git pull` - получение изменений из удаленного репозитория, `git branch` - создание, удаление и переключение веток, `git merge` - слияние изменений из других веток.
- 8) Примеры использования при работе с локальным и удаленным репозиториями включают создание нового репозитория с помощью команды `git init`, клонирование удаленного репозитория с помощью команды `git clone`, от отправку изменений в удаленный репозиторий с помощью команды `git push` и получение изменений из удаленного репозитория с помощью команды `git pull`.
- 9) Ветви (branches) - это механизм, который позволяет создавать отдельные ветки разработки, которые могут быть объединены в основную ветку позже. Ветви могут быть использованы для разработки новых функций, исправления ошибок и тестирования кода.
- 10) Для игнорирования некоторых файлов при коммите можно использовать файл `.gitignore`, в котором перечисляются имена файлов и папок, которые не должны быть добавлены в репозиторий. Файлы и папки, указанные в `.gitignore`, будут проигнорированы при коммите.

6 Выводы

Я изучила идеологию и применение средств контроля версий и освоила умения по работе с git.

Список литературы

Операционные системы