

# **Отчёта по лабораторной работе №2**

Еремина Оксана Андреевна

# 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>8</b>
<b>5</b>	<b>Выводы</b>	<b>12</b>
<b>6</b>	<b>Контрольные вопросы</b>	<b>13</b>
	<b>Список литературы</b>	<b>16</b>

## Список таблиц

## Список иллюстраций

4.1	Установка git и gh . . . . .	8
4.2	Настройка git . . . . .	8
4.3	Создание ключа ssh . . . . .	9
4.4	Создание ключа pgr . . . . .	9
4.5	Добавление ключа pgr . . . . .	10
4.6	Настройка подписей коммитов . . . . .	10
4.7	Настройка gh . . . . .	10
4.8	Создание шаблона курса . . . . .	10
4.9	Настройка каталога курса . . . . .	11

# 1 Цель работы

Целью данной лабораторной работы является освоение практических навыков умений по работе с git

## 2 Задание

1. Создать базовую конфигурацию для работы с git.
2. Создать ключ SSH.
3. Создать ключ PGP.
4. Настроить подписи git.
5. Зарегистрироваться на Github.
6. Создать локальный каталог для выполнения заданий по предмету.

### 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.

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

### 1. Установка программного обеспечения

В терминале ввожу команду, которая устанавливает git и gh(рис.1)

```
oaeremina@oaeremina:~$ sudo apt-get install git
[sudo] пароль для oaeremina:
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Уже установлен пакет git самой новой версии (1:2.34.1-1ubuntu1.10).
Следующие пакеты устанавливались автоматически и больше не требуются:
  libappstream-glib8 libflashrom1 libftdi1-2 libfuse2 liblvm13
для их удаления используйте «sudo apt autoremove».
Обновлено 0 пакетов, установлено 0 новых пакетов, для удаления отмечено 0 пакетов,
и 20 пакетов не обновлено.
oaeremina@oaeremina:~$ sudo apt-get install gh
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Уже установлен пакет gh самой новой версии (2.4.0+dfsg1-2).
Следующие пакеты устанавливались автоматически и больше не требуются:
  libappstream-glib8 libflashrom1 libftdi1-2 libfuse2 liblvm13
для их удаления используйте «sudo apt autoremove».
Обновлено 0 пакетов, установлено 0 новых пакетов, для удаления отмечено 0 пакетов,
и 20 пакетов не обновлено.
```

Рис. 4.1: Установка git и gh

### 2. Базовая настройка git

В терминале задаю имя и email моего репозитория. Далее настраиваю его вводя команды. (рис.2)

```
oaeremina@oaeremina:~$ git config --global user.name "oaeremina"
oaeremina@oaeremina:~$ git config --global user.email "1132236056@pfur.ru"
oaeremina@oaeremina:~$ git config --global core.quotepath false
oaeremina@oaeremina:~$ git config --global init.defaultBranch master
oaeremina@oaeremina:~$ git config --global core.autocrlf input
oaeremina@oaeremina:~$ git config --global core.safecrlf warn
```

Рис. 4.2: Настройка git



### 3. Создание ключа ssh

Генерирую ключ и выбираю нужные опции (рис. 3)

```
oaeremina@oaeremina:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/oaeremina/.ssh/id_rsa):
/home/oaeremina/.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 /home/oaeremina/.ssh/id_rsa
Your public key has been saved in /home/oaeremina/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Fmc6GT0x3MLLFuK0TIKRK4igdtLVG5i98Ley2w4sohA oaeremina@oaeremina
The key's randomart image is:
+---[RSA 4096]-----+
|  .. = o . |
| . .o = * * o |
| = o.o + % O |
| =o.o + * & . |
| E.o   o S o |
| .   .o o |
| . . . oo |
| . . . .o |
| .   .oo |
| .   .oo |
+---[SHA256]-----+
```

Рис. 4.3: Создание ключа ssh

### 4. Создание ключа pgp

Вывожу список ключей и копирую сгенерированный ключ в буфер обмена (рис.4)

```
oaeremina@oaeremina:~$ 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
```

Рис. 4.4: Создание ключа pgp

Перехожу в свой репозиторий, переходу в New GPG key и вставляю код из буфера обмена (рис.5)

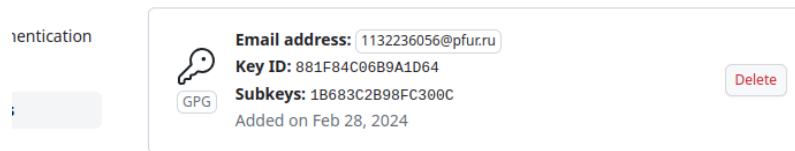


Рис. 4.5: Добавление ключа pgp

## 5. Настройка автоматических подписей коммитов git

Ввожу команды, чтобы git применял мой email при подписи коммитов (рис.6)

```
oaeremina@oaeremina:~$ git config --global user.signingkey 7C7988706565C83C
oaeremina@oaeremina:~$ git config --global commit.gpgsign true
oaeremina@oaeremina:~$ git config --global gpg.program $(which gpg2)
```

Рис. 4.6: Настройка подписей коммитов

## 6. Настройка gh

Авторизуюсь, вводя команду, далее отвечаю на вопросы (рис.7) После ответов на вопросы меня переносит в браузер и я ввожу код, который написан в терминале.

```
oaeremina@oaeremina:~$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: ED9D-0993
- Press Enter to open github.com in your browser...
find_ffmpeg failed, using the integrated library.
Окно или вкладка откроются в текущем сеансе браузера.
```

Рис. 4.7: Настройка gh

## 7. Создание репозитория курса на основе шаблона

Создаю шаблон рабочего пространства, ввожу нужные команды (рис.8)

```
oaeremina@oaeremina:~$ mkdir -p ~/work/study/2023-2024/"Операционные системы"
oaeremina@oaeremina:~$ cd ~/work/study/2023-2024/"Операционные системы"
oaeremina@oaeremina:~/work/study/2023-2024/Операционные системы$ gh repo create
study_2023-2024_os-intro --template=yanadharma/course-directory-student-template
--public
✓ Created repository oaeremina/study_2023-2024_os-intro on GitHub
```

Рис. 4.8: Создание шаблона курса

Перехожу в каталог курса, удаляю лишние файлы, затем создаю необходимые каталоги (рис.9) И отправляю эти файлы на сервер

```
oaeremina@oaeremina:~/work/study/2022-2023/Операционные системы$ cd ~/work/study/2022-2023/Операционные системы/os-intro
oaeremina@oaeremina:~/work/study/2022-2023/Операционные системы/os-intro$ rm package.json
oaeremina@oaeremina:~/work/study/2022-2023/Операционные системы/os-intro$ echo os-intro > COURSE
oaeremina@oaeremina:~/work/study/2022-2023/Операционные системы/os-intro$ make
Usage:
  make <target>

Targets:
  list           List of courses
  prepare       Generate directories structure
  submodule     Update submodules
```

Рис. 4.9: Настройка каталога курса

## 5 Выводы

При выполнении данной лабораторной работы я приобрела практические навыки работы с git

## 6 Контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначены?

Системы контроля версий (VCS) - программное обеспечение для облегчения работы с изменяющейся информацией. Они позволяют хранить несколько версий изменяющейся информации, одного и того же документа, может предоставить доступ к более ранним версиям документа. Используется для работы нескольких человек над проектом, позволяет посмотреть, кто и когда внес какое-либо изменение и т. д. VCS применяются для: Хранения полной истории изменений, сохранения причин всех изменений, поиска причин изменений и совершивших изменение, совместной работы над проектами.

2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочий каталог.

Хранилище – репозиторий, хранилище версий, в нем хранятся все документы, включая историю изменений.

3. Что представляют собой и чем отличаются централизованные и децентрализованные системы контроля версий?

Централизованные VCS (например: CVS, TFS, AccuRev) – одно основное хранилище всей информации.

4. Опишите действия с VCS при единоличной работе с хранилищем.

Сначала создается и подключается удаленный репозиторий, затем по мере изменения проекта локальный репозиторий синхронизируется с удаленным.

5. Опишите порядок работы с общим хранилищем VCS.

Участник проекта перед началом работы получает нужную ему версию проекта в хранилище.

6. Каковы основные задачи, решаемые инструментальным средством git?

Хранение информации о всех изменениях в вашем коде, обеспечение удобства командной работы.

7. Назовите и дайте краткую характеристику командам 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` Просмотр текущих изменений: `git diff` Сохранение текущих изменений: добавить все изменённые и/или созданные файлы и/или каталоги: `git add .` добавить конкретные изменённые и/или созданные файлы и/или каталоги: `git add имена_файлов` удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории): `git rm имена_файлов` Сохранение добавленных изменений: сохранить все добавленные изменения и все изменённые файлы: `git commit -am 'Описание коммита'` сохранить добавленные изменения с внесением комментария через встроенный редактор: `git commit` создание новой ветки, базирующейся на текущей: `git checkout -b имя_ветки` переключение на некоторую ветку: `git checkout имя_ветки` (при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой) отправка изменений конкретной ветки в центральный репозиторий: `git push origin имя_ветки` слияние ветки с текущим деревом: `git merge --no-ff имя_ветки` Удаление ветки: удаление локальной уже слитой с основным деревом ветки: `git branch -d имя_ветки` принудительное уда-

удаление локальной ветки: `git branch -D имя_ветки` удаление ветки с центрального репозитория: `git push origin :имя_ветки`

8. Приведите примеры использования при работе с локальным и удалённым репозиторием

`git push -all` отправляем из локального репозитория все сохраненные изменения в центральный

9. Что такое и зачем могут быть нужны ветви (branches)?

Ветвление - один из параллельных участков в одном хранилище, исходящих из одной ветки

10. Как и зачем можно игнорировать некоторые файлы при commit?

Во время работы над проектом могут создаваться файлы, которые не следуют добавлять в репозиторий

## **Список литературы**

1. Лабораторная работа № 2 URL: <https://esystem.rudn.ru/mod/page/view.php?id=1098790>