

# Отчёт по лабораторной работе №2

дисциплина: Операционные системы

Студент: Махорин Иван Сергеевич

# Содержание

Цель работы	5
Задание	6
Выполнение лабораторной работы	7
Контрольные вопросы	15
Выводы	17

## Список иллюстраций

0.1	Настройка github . . . . .	7
0.2	Установка git-flow в Fedora Linux . . . . .	8
0.3	Установка git-flow в Fedora Linux . . . . .	8
0.4	Базовая настройка git . . . . .	8
0.5	Создание ключа SSH . . . . .	9
0.6	Ввод команды и получение ключа в терминале . . . . .	9
0.7	Добавление нового ключа на github . . . . .	10
0.8	Ключ создан . . . . .	10
0.9	Настройка gh . . . . .	11
0.10	Создание репозитория . . . . .	12
0.11	Создание репозитория . . . . .	12
0.12	Настраиваем каталог курса . . . . .	13
0.13	Создаём необходимые каталоги . . . . .	13
0.14	Отправляем наши файлы на сервер . . . . .	14

## Список таблиц

## Цель работы

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

# 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.

# Выполнение лабораторной работы

Произведём настройку github. Для этого создадим учётную запись на <https://github.com>. После чего заполняем основные данные в профиле. (рис. [-@fig:001])

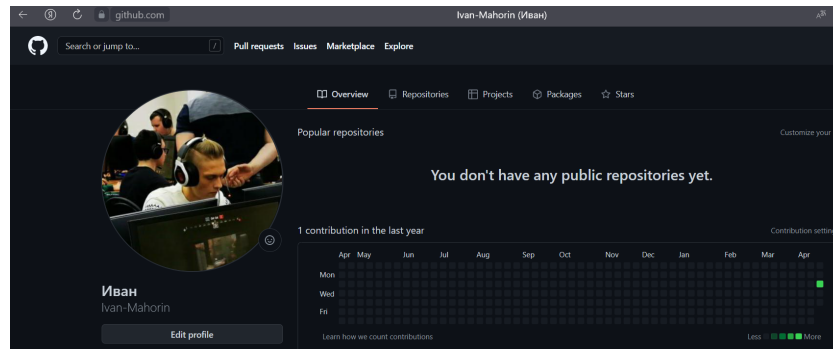


Рис. 0.1: Настройка github

Следующим шагом устанавливаем git-flow в Fedora Linux, так как это программное обеспечение удалено из репозитория. (рис. [-@fig:002]) и (рис. [-@fig:003]).

Команды для установки:

1. `cd /tmp`

```
wget --no-check-certificate -q https://raw.githubusercontent.com/petervanderdoes/gitflow/develop/contrib/gitflow-installer.sh
```

2. `chmod +x gitflow-installer.sh`

3. `sudo ./gitflow-installer.sh install stable`

```
[ismahorin@fedora Операционные системы]$ cd /tmp
```

Рис. 0.2: Установка git-flow в Fedora Linux

```
[ismahorin@fedora tmp]$ wget --no-check-certificate -q https://raw.githubusercontent.com/etervanderdoes/gitflow/develop/contrib/gitflow-installer.sh
[ismahorin@fedora tmp]$ chmod +x gitflow-installer.sh
[ismahorin@fedora tmp]$ sudo ./gitflow-installer.sh instal stable

Мы полагаем, что ваш системный администратор изложил вам основы
безопасности. Как правило, всё сводится к трём следующим правилам:

    №1) Уважайте частную жизнь других.
    №2) Думайте, прежде что-то вводить.
    №3) С большой властью приходит большая ответственность.

[sudo] пароль для ismahorin:
```

Рис. 0.3: Установка git-flow в Fedora Linux

Базовая настройка git (рис. [-@fig:004]):

1. Задаём имя и email владельца репозитория (1 и 2 строка на рисунке)
2. Настраиваем utf-8 в выводе сообщений git (3 строка на рисунке)
3. Настраиваем верификацию и подписание коммитов git. Зададим имя начальной ветки (будем называть её master) (4 строка на рисунке)
4. Параметр autocrlf (5 строка на рисунке)
5. Параметр safecrlf (6 строка на рисунке)

```
[ismahorin@fedora ~]$ git config --global user.name "Ivan-Mahorin"
[ismahorin@fedora ~]$ git config --global user.email "ivan.mahorin@yandex.ru"
[ismahorin@fedora ~]$ git config --global core.quietpath false
[ismahorin@fedora ~]$ git config --global init.defaultBranch master
[ismahorin@fedora ~]$ git config --global core.autocrlf input
[ismahorin@fedora ~]$ git config --global core.safecrlf warn
```

Рис. 0.4: Базовая настройка git

Создаём ключ SSH (рис. [-@fig:005]). В терминале вводим данную команду:  
ssh-keygen -t rsa -b 4096

Далее во всех пунктах пользуемся клавишей Enter и получаем наш ключ.



```
[ismahorin@fedora ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ismahorin/.ssh/id_rsa):
Created directory '/home/ismahorin/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ismahorin/.ssh/id_rsa
Your public key has been saved in /home/ismahorin/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:9R9LxWAUacczBW69I88jRq9JZ5+tG0hhs1fRsEZzbQ ismahorin@fedora
The key's randomart image is:
+---[RSA 4096]-----+
|
|      +*OX|
|      .B=|
|      . +E=|
|      . .. =..|
|      S   ..=o |
|      B X=|
|      . #oB|
|      = @.|
|      ..*..|
+-----[SHA256]-----+
```

Рис. 0.5: Создание ключа SSH

Чтобы можно было скопировать наш ключ и добавить его на github, потребуется команда:

cat ~/ .ssh/id\_rsa/pub (рис. [-@fig:006]).

```
[ismahorin@fedora ~]$ cat ~/ .ssh/id_rsa.pub
cat: /home/ismahorin/: Это каталог
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQACQDERZg8hxbMjF8m5UdRB1s/dgJdXlDsLM+dps+oq60
wxew94YBUQ24bQ567SCkZ07+1bDvRBZemyP4ZZvmP6aVagm58vRVsx4vox+FBs2XeibScE3IoR4qLIK
AnlkZQ54j1WCeBRqFB5Qg1C+ir9mt2Ux3lVnIQIr4TqwEuJvVLZ6QZ9VB1cNDffg+8k79S+KfIgHSfJ
UbRRo5iZ2baNltu06exwE6LjyXqwWftwAwjNWN7Gg7xjh9Md0dvl8xtX03PQ0JzP26Q95398YwdjrEv
a1LJmkeF6pzA9CCpUUxtgFbOwpcMr+irYsXlTl/9ohFr+Q/VDt6psGSpgPa1A1mPHTuVJ9Lwc3bh54r
Fb/6kI31JobTHyKKKm0U3ET4JcPgS8gLUXq1qMMKqG5xs1/eD7tAtY8r8ssHPuIzd3ig/mas60wmPo
4nz8LuU2+LUNzf8dIhqv5+BSX0pGSqVLkcV8rdySu/L1U/tlb2I/uZ7QH3UrxafItJd53H8vN3yqVpz
xrhL7YsuK86DceNsMrvBbdQsqEIwft3h7gzAwf+/DHDF+T9jTH6hg2IM9V6KJfkEt5kYJrzyjeQqPgH
fmLX3lckuvJ0P/p8oAjC8VKRLkcXhMnks9XlAMzRkrVuv0UalqDEggC2xYiBDb0SQ//gVU1yJGITGmo
z39uDUR4p7Q== ismahorin@fedora
```

Рис. 0.6: Ввод команды и получение ключа в терминале

Скопированный ключ нужно добавить на github. Для этого переходим на сайте в раздел “Settings” и выбираем “SSH and GPG keys” (рис. [-@fig:007]). Как только мы выбрали SHH keys, даём название нашему ключу и копируем ключ из терминала.

В моём случае был создан ключ под названием OS (операционные системы) (рис. [-@fig:008]).

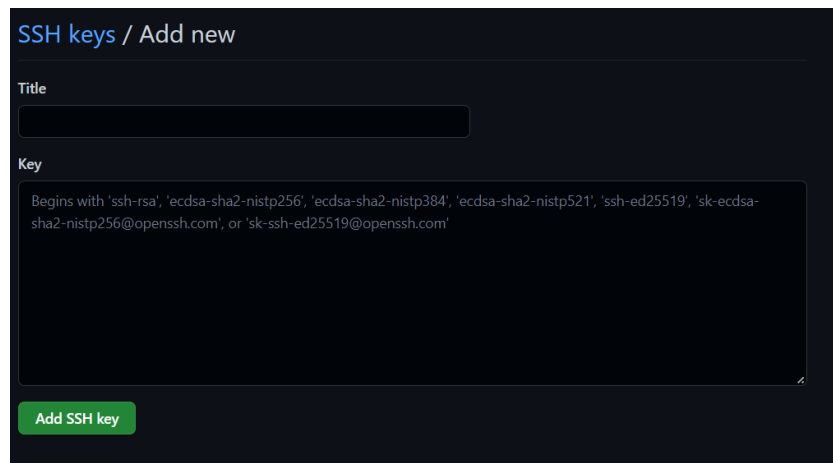


Рис. 0.7: Добавление нового ключа на github

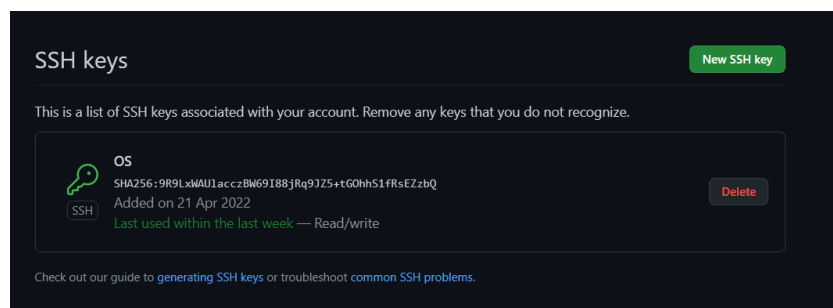


Рис. 0.8: Ключ создан

Возвращаемся в наш терминал и настраиваем gh командой:

gh auth login (рис. [-@fig:009]).

Во всех пунктах выбираем у(yes).

По полученной ссылке переходим в браузер на виртуальной машине и вводим код из терминала (находится перед ссылкой).

```

[ismahorin@fedora ~]$ gh auth login
bash: gh: command not found...
Install package 'gh' to provide command 'gh'? [N/y] y

* Waiting in queue...
* Loading list of packages...
The following packages have to be installed:
gh-2.7.0-1.fc35.x86_64 GitHub's official command line tool
Proceed with changes? [N/y] y

* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...

! First copy your one-time code: 8017-949D
Open this URL to continue in your web browser: https://github.com/login/device

```

Рис. 0.9: Настройка gh

Создаём репозиторий курса на основе шаблона (рис. [-@fig:010]) и (рис. [-@fig:011]). Все нужные команды для создания были в указаниях к лабораторной работе. В 4 команде, вместо , указываем своё имя профиля на github.

1. `mkdir -p ~/work/study/2021-2022/“Операционные системы”`
2. `cd ~/work/study/2021-2022/“Операционные системы”`
3. `gh repo create study_2021-2022_os-intro --template=yamadharm/course-directory-student-template --public`
4. `git clone --recursive git@github.com:/study_2021-2022_os-intro.git os-intro`

```
[ismahorin@fedora ~]$ mkdir -p ~/work/study/2021-2022/"Операционные системы"
[ismahorin@fedora ~]$ cd ~/work/study/2021-2022/"Операционные системы"
[ismahorin@fedora Операционные системы]$ gh repo create study_2021-2022_os-intro
o`--public`, `--private`, or `--internal` required when not running interactive
ly

Usage:  gh repo create [<name>] [flags]

Flags:
  -c, --clone                Clone the new repository to the current directory
  -d, --description string   Description of the repository
      --disable-issues       Disable issues in the new repository
      --disable-wiki         Disable wiki in the new repository
  -g, --gitignore string     Specify a gitignore template for the repository
  -h, --homepage URL         Repository home page URL
      --internal             Make the new repository internal
  -l, --license string       Specify an Open Source License for the repository
      --private              Make the new repository private
      --public               Make the new repository public
      --push                 Push local commits to the new repository
  -r, --remote string        Specify remote name for the new repository
  -s, --source string        Specify path to local repository to use as source
```

Рис. 0.10: Создание репозитория

```
[ismahorin@fedora Операционные системы]$ git clone --recursive git@github.com:I
van-Mahorin/study_2021-2022_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:+DiY3wvV6TuJJhpZisF/zLDA0zPMSvHdKr4UvC0qU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 20, done.
remote: Counting objects: 100% (20/20), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 20 (delta 2), reused 15 (delta 2), pack-reused 0
Получение объектов: 100% (20/20), 12.50 КиБ | 12.50 МБ/с, готово.
Определение изменений: 100% (2/2), 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/ismahorin/work/study/2021-2022/Операционные системы/os-intro/template/presentation»...
remote: Enumerating objects: 42, done.
remote: Counting objects: 100% (42/42), done.
remote: Compressing objects: 100% (34/34), done.
```

Рис. 0.11: Создание репозитория

Настраиваем каталог курса (рис. [-@fig:012]). Для этого переходим в него командой:

```
cd ~/work/study/2021-2022/"Операционные системы"/os-intro
```

Далее командой `ls` проверяем, что мы в него перешли. В каталоге “os-intro” нам потребуется удалить файл “package.json”. Выполняем данную задачу командой:

```
rm package.json
```

Снова командой `ls` проверяем успешное выполнение удаление файла.

```
[ismahorin@fedora Операционные системы]$ cd ~/work/study/2021-2022/"Операционны
е системы"/os-intro
[ismahorin@fedora os-intro]$ ls
config  Makefile      README.en.md  README.md
LICENSE package.json  README.git-flow.md  template
[ismahorin@fedora os-intro]$ rm package.json
[ismahorin@fedora os-intro]$ ls
config  Makefile      README.git-flow.md  template
LICENSE README.en.md  README.md
```

Рис. 0.12: Настраиваем каталог курса

Создаём необходимые каталоги и отправляем наши файлы на сервер (рис. [-@fig:013]) и (рис. [-@fig:014]).

`make COURSE=os-intro`

1. `git add .`
2. `git commit -am 'feat(main): make course structure'`
3. `git push`

```
[ismahorin@fedora os-intro]$ make COURSE=os-intro
[ismahorin@fedora os-intro]$ git add .
[ismahorin@fedora os-intro]$ git commit -am 'feat(main): make course structure'
[master 238767d] feat(main): make course structure
149 files changed, 16590 insertions(+), 14 deletions(-)
create mode 100644 labs/lab01/presentation/Makefile
create mode 100644 labs/lab01/presentation/presentation.md
create mode 100644 labs/lab01/report/Makefile
create mode 100644 labs/lab01/report/bib/cite.bib
create mode 100644 labs/lab01/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab01/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 labs/lab01/report/report.md
create mode 100644 labs/lab02/presentation/Makefile
create mode 100644 labs/lab02/presentation/presentation.md
create mode 100644 labs/lab02/report/Makefile
create mode 100644 labs/lab02/report/bib/cite.bib
create mode 100644 labs/lab02/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab02/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 labs/lab02/report/report.md
create mode 100644 labs/lab03/presentation/Makefile
create mode 100644 labs/lab03/presentation/presentation.md
create mode 100644 labs/lab03/report/Makefile
create mode 100644 labs/lab03/report/bib/cite.bib
```

Рис. 0.13: Создаём необходимые каталоги

```
[ismahorin@fedora os-intro]$ git push
Перечисление объектов: 20, готово.
Подсчет объектов: 100% (20/20), готово.
Сжатие объектов: 100% (16/16), готово.
Запись объектов: 100% (19/19), 265.88 КиБ | 2.20 МиБ/с, готово.
Всего 19 (изменений 2), повторно использовано 0 (изменений 0), повторно использо
вано пакетов 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To github.com:Ivan-Mahorin/study_2021-2022_os-intro.git
   6683fd4..238767d  master -> master
```

Рис. 0.14: Отправляем наши файлы на сервер

# Контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются? Это программное обеспечение для облегчения работы с изменяющейся информацией. VCS позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое.
2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия. Хранилище (repository), или репозиторий, — место хранения всех версий и служебной информации. Commit («[трудовой] вклад», не переводится) — синоним версии; процесс создания новой версии. История — место, где сохраняются все коммиты, по которым можно посмотреть данные о коммитах. Рабочая копия — текущее состояние файлов проекта, основанное на версии, загруженной из хранилища.
3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида. Централизованные VCS: одно основное хранилище всего проекта и каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно. Децентрализованные VCS: у каждого пользователя свой вариант (возможно не один) репозитория.
4. Опишите действия с VCS при единоличной работе с хранилищем.
5. Опишите порядок работы с общим хранилищем VCS.
6. Каковы основные задачи, решаемые инструментальным средством git? Git — это система управления версиями. У Git две основных задачи: первая —

хранить информацию о всех изменениях в вашем коде, начиная с самой первой строки, а вторая — обеспечение удобства командной работы над кодом.

7. Назовите и дайте краткую характеристику командам `git`. `git --version` (Проверка версии Git) `git init` (Инициализировать ваш текущий рабочий каталог как Git-репозиторий) `git clone https://www.github.com/username/repo-name` (Скопировать существующий удаленный Git-репозиторий) `git remote` (Просмотреть список текущих удалённых репозиториях Git) `git remote -v` (Для более подробного вывода) `git add my_script.py` (Можете указать в команде конкретный файл). `git add .` (Позволяет охватить все файлы в текущем каталоге, включая файлы, чье имя начинается с точки) `git commit -am "Commit message"` (Вы можете сжать все индексированные файлы и отправить коммит). `git branch` (Просмотреть список текущих веток можно с помощью команды `branch`) `git --help` (Чтобы узнать больше обо всех доступных параметрах и командах) `git push origin master` (Передать локальные коммиты в ветку удаленного репозитория).
8. Приведите примеры использования при работе с локальным и удалённым репозиториями.
9. Что такое и зачем могут быть нужны ветки (`branches`)? Ветки нужны, чтобы несколько программистов могли вести работу над одним и тем же проектом или даже файлом одновременно, при этом не мешая друг другу. Кроме того, ветки используются для тестирования экспериментальных функций: чтобы не повредить основному проекту, создается новая ветка специально для экспериментов.
10. Как и зачем можно игнорировать некоторые файлы при `commit`? Игнорируемые файлы — это, как правило, артефакты сборки и файлы, генерируемые машиной из исходных файлов в вашем репозитории, либо файлы, которые по какой-либо иной причине не должны попадать в коммиты.



## Выводы

В ходе выполнения лабораторной работы изучили идеологию и применение средств контроля версий, а также освоили умения по работе с git.