

Лабораторная работа 2

Руслан Исмаилов Шухратович

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Контрольные вопросы	12
5	Выводы	17

Список иллюстраций

3.1	установка gh, git	7
3.2	примеры выполненных команд	8
3.3	ssh ключ	8
3.4	gpg ключ	9
3.5	привязанные ключи	10
3.6	выполненные команды для создания репозитория	10
3.7	Страница github	11

Список таблиц

1 Цель работы

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

2 Задание

- Создать базовую конфигурацию для работы с git.
- Создать ключ SSH.
- Создать ключ PGP.
- Настроить подписи git.
- Зарегистрироваться на Github.
- Создать локальный каталог для выполнения заданий по предмету.

3 Выполнение лабораторной работы

Начнем с установки git и gh, требуемых для выполнения последующей работы. Это можно сделать при помощи команды `sudo dnf install` (рис. 3.1).

```
[rsismailov@fedora ~]$ sudo dnf install git
[sudo] пароль для rsismailov:
Copr repo for PyCharm owned by phracek      5.4 kB/s | 3.6 kB    00:00
Copr repo for PyCharm owned by phracek      25 kB/s | 44 kB    00:01
Fedora 36 - x86_64                          19 kB/s | 18 kB    00:00
Fedora 36 openh264 (From Cisco) - x86_64    903 B/s | 989 B    00:01
Fedora Modular 36 - x86_64                  16 kB/s | 18 kB    00:01
Fedora 36 - x86_64 - Updates                 17 kB/s | 12 kB    00:00
Fedora 36 - x86_64 - Updates                 639 kB/s | 13 MB   00:21
Fedora Modular 36 - x86_64 - Updates         21 kB/s | 18 kB    00:00
Fedora Modular 36 - x86_64 - Updates        112 kB/s | 114 kB  00:01
google-chrome                               7.4 kB/s | 1.3 kB   00:00
google-chrome                               7.0 kB/s | 3.6 kB   00:00
RPM Fusion for Fedora 36 - Nonfree - NVIDIA Dri 4.9 kB/s | 7.5 kB   00:01
RPM Fusion for Fedora 36 - Nonfree - NVIDIA Dri 31 kB/s | 15 kB    00:00
RPM Fusion for Fedora 36 - Nonfree - Steam    7.1 kB/s | 6.7 kB   00:00
Пакет git-2.35.1-1.fc36.x86_64 уже установлен.
Зависимости разрешены.
Отсутствуют действия для выполнения.
Выполнено!
[rsismailov@fedora ~]$ sudo dnf install gh
Последняя проверка окончания срока действия метаданных: 0:00:26 назад, Пт 17 фев 2023 19:17:00.
Зависимости разрешены.
=====
Пакет      Архитектура      Версия      Репозиторий      Размер
=====
Установка:
gh          x86_64           2.22.1-1.fc36      updates          8.2 М
=====
Результат транзакции
=====
Установка 1 Пакет

Объем загрузки: 8.2 М
Объем изменений: 41 М
Продолжить? [д/н]: y
Загрузка пакетов:
gh-2.22.1-1.fc36.x86_64.rpm                                686 kB/s | 8.2 MB    00:12
-----
Общий размер                                              669 kB/s | 8.2 MB    00:12
Проверка транзакции
Проверка транзакции успешно завершена.
Идет проверка транзакции
Тест транзакции проведен успешно.
Выполнение транзакции
Подготовка :
Установка : gh-2.22.1-1.fc36.x86_64                      1/1
Запуск скриптлеты: gh-2.22.1-1.fc36.x86_64                1/1
```

Рис. 3.1: установка gh, git

Далее требуется провести базовую настройку, а именно:
установить имя и почту пользователя
Настройка utf-8 в выводе сообщений git

Зададим имя начальной ветки
и прочее.
(рис. 3.2).

```
Выполнено!  
[rsismailov@fedora ~]$ git config --global user.name "Rsismailov"  
[rsismailov@fedora ~]$ git config --global user.email "Woodentoasterlol@gmail.com"  
[rsismailov@fedora ~]$ git config --global core.quotePath false
```

Рис. 3.2: примеры выполненных команд

Нам требуется сгенерировать ключи для использования github:
Ssh
(рис. 3.3).

```
[rsismailov@fedora ~]$ ssh-keygen -t rsa -b 4096  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/rsismailov/.ssh/id_rsa):  
/home/rsismailov/.ssh/id_rsa already exists.  
Overwrite (y/n)? n  
[rsismailov@fedora ~]$ ssh-keygen -t ed25519  
Generating public/private ed25519 key pair.  
Enter file in which to save the key (/home/rsismailov/.ssh/id_ed25519):  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/rsismailov/.ssh/id_ed25519  
Your public key has been saved in /home/rsismailov/.ssh/id_ed25519.pub  
The key fingerprint is:  
SHA256:tWS/zP1xqRZOUapbDfoqY/CuXL9y5dhlb0l18z0JUI4 rsismailov@fedora  
The key's randomart image is:  
+--[ED25519 256]--+  
|      ..      |  
|      .o .    |  
|     +E..o    |  
|      + o = .o |  
|      S . + = B|  
|      . = B B+ |  
|      o. # *o= |  
|      . *.+ *.o=|  
|      o+.=++ . .|  
+----[SHA256]-----+
```

Рис. 3.3: ssh ключ

Pgp
(рис. 3.4).


```

rsismailov@fedora ~]$ gpg --armor --export 41266CE3AB9D98331B78201B0B64B3DF012CBEA1
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGpVqZ4BEADcIdPRjyE0aF4qJs+EapCxiBGtl+eljcn/fwPoSuvzXooKhqk3
+h91YvtiZRCRdcqs99MN2zctrJX08VxdoJ0jKJmi9SCZuQpffetgdM4rMzPvoi
/s7ylvXzLepjj9/+SYFKdByrK5aMl0nOgImKjDZ32o3VrUtJ8sUksxZ4uHLe8U0x
C2d87BXDCxxQKT9r164JsnXMLFQIZoekqF0SEEIgCTA5zgYyaJJr03feTJLbfKav
8mMAMoxsGBwxHSDnEeXBMHCXEZSUG5CFkNIbNoqDIJUUVIJEN0BaDbb11K/LAesDd
uRg0WqrUn8txGhNfMmFD40BC9V+W7XF8fADpHSJ0wc6fwxGtxKwphL9xgmbL90
KA3jhm3UdmgjHNSZnF2l+tWAUDjFcWF7bB8sTGvzIwOH7Za99pSfXT2Z5AnBSuNj
pE7IGs9BI1bi/i9G1uAbRe2c07u1DLV6yohIu11UvHqm8s6K8Hd0a6M5tIbN+yG
/iUcy6NyoozYnYYLd/vTvT5HHl7WBuCSsXn+RbiChg9Ed3h9HGouGno/hQ9mn0IX
L462WMVo3+BnsBfj7L/NOpceQnQoVvYyZ2WNKXbG49d6BwKT/7tSZbsagqKpo6Pq
2j08BIV0Xdrb7NaTmpl7X43Usq5WYnXZ5CZ5Wi9oZWkIv0BymayS9JYgewARAB
DrQoNGD0YHQu9Cw0L0g0JjRgdC80LDQuNC70L7QsIA8d29vZGVud69hc3Rlcmxv
pEBnbWFPbc5jb20+iQJ5BBMBCAA8FiEEQSZs46udmDMbeCAbC2S3wEsvqEFAmPv
qZ4CGwMFCwkIBwIDIgIBBhUKCQgLAgQwAqMBAh4HAheAAAJEAtks98BLL6HuZYQ
AKr24L1AiFVXIYVQU49KBsDbkxX0bcDNKIab6xdAB2ms2dc5eDT4vwZ7TWCuc91G
Y2U0ZlI2Z5HgLoURJoVsoJNUPRTsaIk3rWxNMwgWmiM6ZLv+uLfdfronC5dfBntJh
0mCvYiaYpYdpDvHGFaSQ3jf5Z1xtDU7Q7PJe94/jEXRUGNnf6TgpuQw2x5gwRD
7S50nPkrMsMjM3Cj3CBASR8fqlNFXWoZ3K0vLn662Kf6NnY/9fqJiQ3+exqLVIOr
eSDFFoLM00YCPouRu8iuDSLnz+eT+x8lPmQB466ZsrgyAjJuc3d7yYowb4GikhDH
0fqXhBg3HkR3UfsBD2G12wi1z8bg00L6ZB6rNPvRDsra2LM7MQ0wN50+fLeVYcMt
L8ECTqH8kvVl+G3UDZ43/nriGw7azZ/LG3uJhp+pWdFQpda25wjLDly8IgDlwUP/
XJ/Sxhqsngk64cnuYjiJquFcy2wXbpltrsSVHhjUViH9xotTrqhKPM/wULH506Q
F4UVn21kdTKpV3u6GvpECgX4ReQNV6g/blGN+J09Emy3zjZoCC1wEqImk0ag3b7H
0DdLKUE4dTs6BtLzmzYSY37000o6Hu49WwtpE3Vr3Pc2Qdw03Y5CilrZfZAddrRL
eVe+9jFDicqn9AdoNdiB8G0oTI5hx7rJtB4SibixvP+uQINBGpVqZ4BEACuzFQM
pHYEaDhkyIZqBnUovhBilnqBViLmN9nup7z8IAQka7afBUMRzB5HaqRodHiRtEBt
0A6ujQPslo/02kPanuM2a45AyHPLxMdoGEsXL4f87Gop0R3P/gjFhmIGroQJMjZp
08G8U7tyYrbSHXk20+II/fQBewtXpQkF5UxJ2dpXJZrpEhgxEaQxUM0we5YVpAP+
C44HQc4InB8/m02e84guygSWKbiPS5pmGmbAvF81C9EiIDasVv2ayLxhu142Tsyd
/XrHKBxzRQ0KEiICvS/h8Ytu+/m3EpruE9sxM5BmDg0sLzP2I+AX3mILYJ7uWrZy
BzV8UDDFRnT9D/93BVh5xHfoZmCGLtv9cKlZfP2WBB86sLNXjwnqsEnynj4wH
/UNZYQv4HdfLhrXDYezEzyBh5FAuQ2cB+j38qBK044nI7i5GZreLb82qF6TNcpmg
kzH+F+V2Z4Iq348RdLLjWJCuu786AnyBh6XeqYvcXhR21fqRVJ2QDnp78b8ynQkJ
0Gfx+wLLJDKVAGOR0p0/SqiQ1JF7pkC9sc62bYviQ45tUu9iXB34PXIOYnZ/aS34

```

Рис. 3.4: gpg ключ

Привязываем их к аккаунту на github.com, после авторизации (можно выпол-
нить через консоль)



(рис. 3.5).

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	Auth SHA256 : x2NbsbtJ0zYUheVDH70eCxxTo+iWEj1VFTIUHAX5ox4 Added on Oct 15, 2022 Last used within the last 2 months — Read/write	Delete
 SSH	GitHub CLI SHA256 : tWS/zP1xqRZ0UapbDfoqY/CuXL9y5dh1b0l18z0JUI4 Added on Feb 17, 2023 by GitHub CLI Never used — Read/write	Delete

Check out our guide to [generating 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	gpg Email address: <code>woodentoasterlol@gmail.com</code> Key ID: <code>0B64B3DF012CBEA1</code> Subkeys: <code>3CE630A26BB7C311</code> Added on Feb 17, 2023	Delete
-------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------

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

Рис. 3.5: привязанные ключи

Создадим репозиторий курса на основе шаблона:
(рис. 3.6).

```
[rsismailov@fedora Операционные системы]$ git clone --recursive git@github.com:rsismailov/study_2022-2023_os-intro.git
Клонирование в «os-intro»...
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.
```

Рис. 3.6: выполненные команды для создания репозитория

Отправим все файлы нового локального каталога курса на github:
(рис. 3.7).

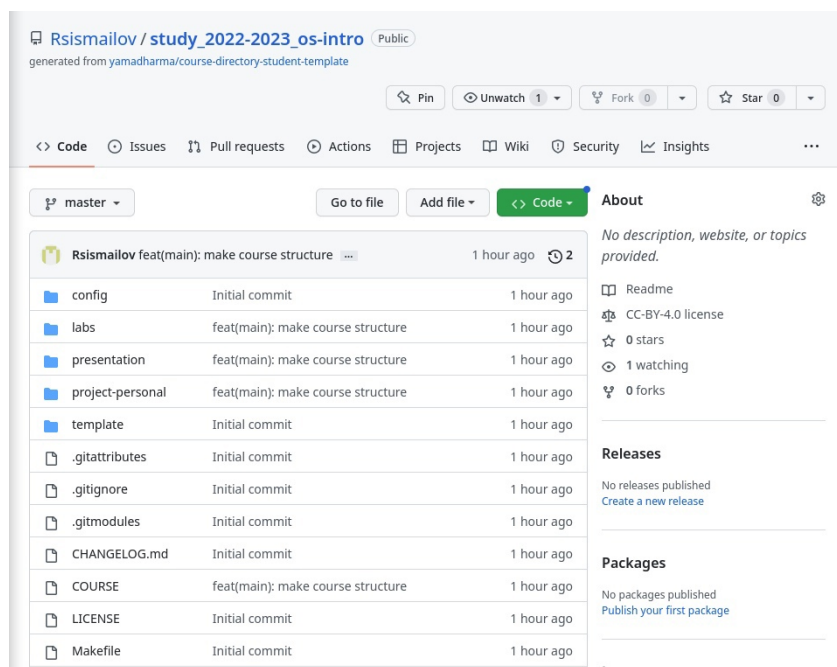


Рис. 3.7: Страница github

4 Контрольные вопросы

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

Контроль версий, также известный как управление исходным кодом, – это практика отслеживания изменений программного кода и управления ими. Системы контроля версий – это программные инструменты, помогающие командам разработчиков управлять изменениями в исходном коде с течением времени.

Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.

Репозиторий – хранилище версий – в нем хранятся все документы вместе с историей их изменения и другой служебной информацией. Рабочая копия – копия проекта, связанная с репозиторием. commit – сохранение изменений в репозитории

Что представляют собой и чем отличаются централизованные и децентрализованные VCS?
Приведите примеры VCS каждого вида.

Централизованные системы контроля версий представляют собой приложения типа клиент-сервер, когда репозиторий проекта существует в единственном экземпляре и хранится на сервере. Доступ к нему осуществлялся через специальное клиентское приложение.

В качестве примеров таких программных продуктов можно привести CVS, Subversion.

Децентрализованные системы контроля версий – СКВ, которые позволяют клиенту полностью хранить у себя копию репозитория проекта.

Примеры: Git, Mercurial, Bazaar или, например, Darcs.

Опишите действия с VCS при единоличной работе с хранилищем.

Хранилище является разновидностью файл-сервера, однако не совсем обычного. •Хранилище запоминает каждое внесенное изменение: -любое изменение любого файла, -изменения в самом дереве каталогов, такие как добавление, удаление и реорганизация файлов и каталогов.

•При чтении данных из хранилища клиент обычно видит только последнюю версию дерева файлов. •Клиент также имеет возможность просмотреть предыдущие состояния файловой системы. •Вопросы типа «Что содержал этот каталог в

прошлую среду?», «Кто был последним, изменявшим этот файл, и какие вносились изменения?»

Опишите порядок работы с общим хранилищем VCS.

Каковы основные задачи, решаемые инструментальным средством git? 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.

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`

Приведите примеры использования при работе с локальным и удалённым репозиториями.

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

Ветки нужны для того, чтобы программисты могли вести

совместную работу над проектом и не мешать друг

другу при этом. При создании проекта, Git создает

базовую ветку. Она называется master веткой.

Как и зачем можно игнорировать некоторые файлы при commit?

Чтобы игнорировать файл, для которого ранее был сделан коммит, необходимо удалить этот файл из репозитория, а затем добавить для него правило в `.gitignore`.

5 Выводы

Я успешно смог применить систему контроля версиями git и создал репозиторий курса на github.com