

Отчёт по лабораторной работе №2

Управление версиями

Джозеф Кервенс

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Вывод	10
4	Контрольные вопросы	11
	Список литературы	15

Список иллюстраций

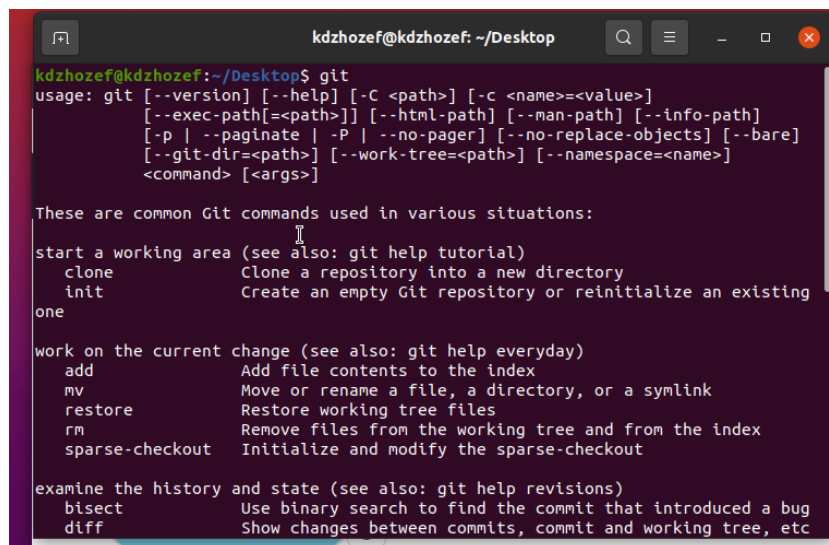
2.1	Загрузка пакетов	5
2.2	Параметры репозитория	5
2.3	rsa-4096	6
2.4	ed25519	6
2.5	GPG ключ	7
2.6	GPG ключ	7
2.7	Параметры репозитория	8
2.8	Связь репозитория с аккаунтом	8
2.9	Загрузка шаблона	9
2.10	Первый коммит	9

1 Цель работы

Целью данной работы является изучение идеологии и применения средств контроля версий и освоение умений работать с git.

2 Выполнение лабораторной работы

Устанавливаем git, git-flow и gh.



```
kdzhozef@kdzhozef:~/Desktop$ git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
      [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
      [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
      [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
      <command> [<args>]

These are common Git commands used in various situations:

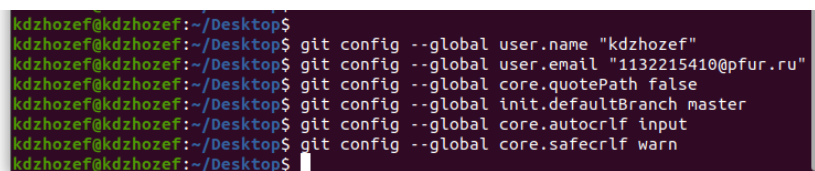
start a working area (see also: git help tutorial)
  clone             Clone a repository into a new directory
  init              Create an empty Git repository or reinitialize an existing
  one

work on the current change (see also: git help everyday)
  add               Add file contents to the index
  mv                Move or rename a file, a directory, or a symlink
  restore            Restore working tree files
  rm                Remove files from the working tree and from the index
  sparse-checkout    Initialize and modify the sparse-checkout

examine the history and state (see also: git help revisions)
  bisect            Use binary search to find the commit that introduced a bug
  diff              Show changes between commits, commit and working tree, etc
```

Рис. 2.1: Загрузка пакетов

Зададим имя и email владельца репозитория, кодировку и прочие параметры.



```
kdzhozef@kdzhozef:~/Desktop$ git config --global user.name "kdzhozef"
kdzhozef@kdzhozef:~/Desktop$ git config --global user.email "1132215410@pfur.ru"
kdzhozef@kdzhozef:~/Desktop$ git config --global core.quotePath false
kdzhozef@kdzhozef:~/Desktop$ git config --global init.defaultBranch master
kdzhozef@kdzhozef:~/Desktop$ git config --global core.autocrlf input
kdzhozef@kdzhozef:~/Desktop$ git config --global core.safecrlf warn
kdzhozef@kdzhozef:~/Desktop$
```

Рис. 2.2: Параметры репозитория

Создаем SSH ключи

```
kdzhofef@kdzhofef: ~/Desktop
kdzhofef@kdzhofef:~/Desktop$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kdzhofef/.ssh/id_rsa):
/home/kdzhofef/.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/kdzhofef/.ssh/id_rsa
Your public key has been saved in /home/kdzhofef/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:+JDnuhzHnhnGm1LMqSoSCyT98Y9LhFaH560cF0EZANK kdzhofef@kdzhofef
The key's randomart image is:
+---[RSA 4096]---+
| .+..oo |
| . Eo. |
| o + |
| . o * . . |
| . o + = S o |
| o o = X * |
| . .B @ o |
| .+o @ O |
| .+BoO |
+-----[SHA256]-----+
kdzhofef@kdzhofef:~/Desktop$
```

Рис. 2.3: rsa-4096

```
kdzhofef@kdzhofef:~/Desktop
kdzhofef@kdzhofef:~/Desktop$
kdzhofef@kdzhofef:~/Desktop$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/kdzhofef/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kdzhofef/.ssh/id_ed25519
Your public key has been saved in /home/kdzhofef/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:8somZkxsWdktspLb4dZlfdEgJW5BWR4C9mEksfoGgzs kdzhofef@kdzhofef
The key's randomart image is:
+--[ED25519 256]--+
|      ==0++ |
|      *O*.. |
|      o .. +..o |
|      +.o... . |
|      . +ooS. . |
|      B o+ +o . |
|      + =Eo.oo . |
|      *. =O.. |
|      o +o |
+-----[SHA256]-----+
kdzhofef@kdzhofef:~/Desktop$
```

Рис. 2.4: ed25519

Создаем GPG ключ

```
kdzhofef@kdzhofef: ~/Desktop
"kdzhofef <1132215410@pfur.ru>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? 0
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /home/kdzhofef/.gnupg/trustdb.gpg: trustdb created
gpg: key 7B0EF281CD051076 marked as ultimately trusted
gpg: directory '/home/kdzhofef/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/kdzhofef/.gnupg/openpgp-revocs.d/75
BDF6BF095FC38D7F2761417B0EF281CD051076.rev'
public and secret key created and signed.

pub  rsa4096 2023-02-16 [SC]
    75BDF6BF095FC38D7F2761417B0EF281CD051076
uid                               kdzhofef <1132215410@pfur.ru>
sub  rsa4096 2023-02-16 [E]

kdzhofef@kdzhofef:~/Desktop$
```

Рис. 2.5: GPG ключ

Добавляем GPG ключ в аккаунт

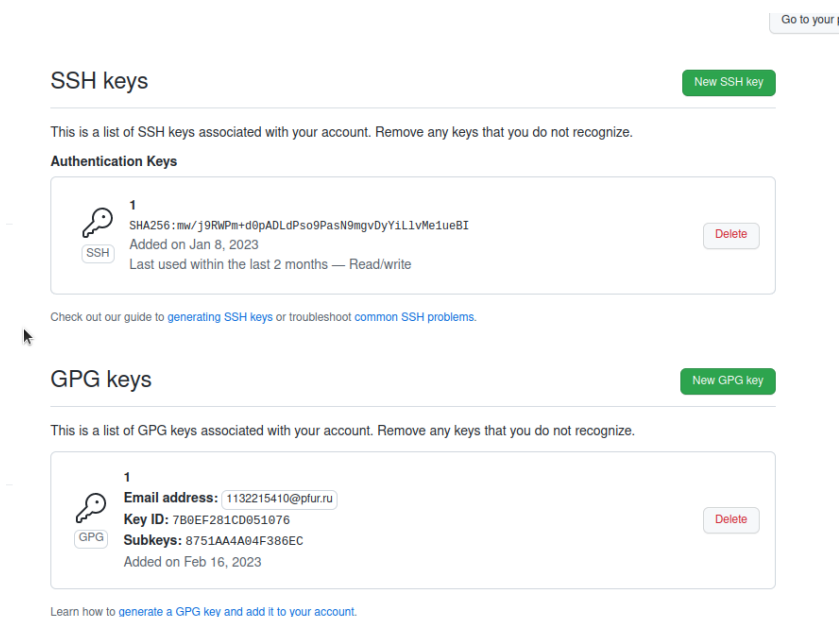


Рис. 2.6: GPG ключ

Настройка автоматических подписей коммитов git

```
kdzhofez@kdzhofez: ~/Desktop
wmfZoYqtAt8XcMI2BXmo+nusgEG7gMQ8GeDQ8ge75eb6GlepkA2gVSQp0xAGg22X
3sLInpSLlpjPCDcPuk2uRCgzdPcOmBeMGfQyhCL7tNqWSnwzKOpUawNemnaM/pBO
RV/tAKGcDXFm/+S3811irbnp1YeAk0MRtp9PxxAgghx99TNaP6iK340HxEHD9Ygz
DpMdNqRCmZnyq9YSLv0uX6I3w1ktuzepf9pTJxf922NIz30D/DBLGulx8DU0JU2w
qz0jrJyv2ZkqwcgCmNAPF0ksA9M1Cyq71g0HLWqy0+VDeIzQ+SSQikAYIm2aP9hY
PxgU1+kuYyEej9/72JV8Fe/FL5Qo1FXckSTvPeRVkKh2iXA2yKxEz9g=
=vW3Y
-----END PGP PUBLIC KEY BLOCK-----
kdzhofez@kdzhofez:~/Desktop$
kdzhofez@kdzhofez:~/Desktop$
kdzhofez@kdzhofez:~/Desktop$
kdzhofez@kdzhofez:~/Desktop$ gpg --list-secret-keys --keyid-format LONG
/home/kdzhofez/.gnupg/pubring.kbx
-----
sec   rsa4096/7B0EF281CD051076 2023-02-16 [SC]
      758DF6BF095FC38D7F2761417B0EF281CD051076
uid    [ultimate] kdzhofez <1132215410@pfur.ru>
ssb    rsa4096/8751AA4A04F386EC 2023-02-16 [E]

kdzhofez@kdzhofez:~/Desktop$ git config --global commit.gpgSign true
kdzhofez@kdzhofez:~/Desktop$ git config --global user.signingKey 7B0EF281CD05107
6
kdzhofez@kdzhofez:~/Desktop$ git config --global gpg.program $(which gpg2)
kdzhofez@kdzhofez:~/Desktop$
```

Рис. 2.7: Параметры репозитория

Настройка gh

```
kdzhofez@kdzhofez: ~/Desktop
ssb   rsa4096/8751AA4A04F386EC 2023-02-16 [E]

kdzhofez@kdzhofez:~/Desktop$ git config --global commit.gpgSign true
kdzhofez@kdzhofez:~/Desktop$ git config --global user.signingKey 7B0EF281CD05107
6
kdzhofez@kdzhofez:~/Desktop$ git config --global gpg.program $(which gpg2)
kdzhofez@kdzhofez:~/Desktop$
kdzhofez@kdzhofez:~/Desktop$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? SSH
? Upload your SSH public key to your GitHub account? /home/kdzhofez/.ssh/id_rsa.
pub
? Title for your SSH key: GitHub CLI
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: E440-4446
Press Enter to open github.com in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
✓ Uploaded the SSH key to your GitHub account: /home/kdzhofez/.ssh/id_rsa.pub
✓ Logged in as kdzhofez
kdzhofez@kdzhofez:~/Desktop$
```

Рис. 2.8: Связь репозитория с аккаунтом

Загрузка шаблона репозитория и синхронизация


```
kdzhofef@kdzhofef: ~/work/study/2022-2023/Операционны...
Submodule 'template/presentation' (https://github.com/yamadharma/academic-presen-
tation-markdown-template.git) registered for path 'template/presentation'
Submodule 'template/report' (https://github.com/yamadharma/academic-laboratory-r
eport-template.git) registered for path 'template/report'
Cloning into '/home/kdzhofef/work/study/2022-2023/Операционные системы/os-intro/
template/presentation'...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Cloning into '/home/kdzhofef/work/study/2022-2023/Операционные системы/os-intro/
template/report'...
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
Receiving objects: 100% (101/101), 327.25 KiB | 3.34 MiB/s, done.
Resolving deltas: 100% (40/40), done.
Submodule path 'template/presentation': checked out 'b1be3800ee91f5809264cb755d3
16174540b753e'
Submodule path 'template/report': checked out '1d1b61dcac9c287a83917b82e3aef11a3
3b1e3b2'
kdzhofef@kdzhofef:~/work/study/2022-2023/Операционные системы$
kdzhofef@kdzhofef:~/work/study/2022-2023/Операционные системы$
```

Рис. 2.9: Загрузка шаблона

Подготовка репозитория и коммит изменений

```
kdzhofef@kdzhofef: ~/work/study/2022-2023/Операционны...
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_secnos.
py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_tableno
s.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/_i
nit_.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/cor
e.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/mai
n.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/pan
docattributes.py
create mode 100644 project-personal/stage6/report/report.md
kdzhofef@kdzhofef:~/work/study/2022-2023/Операционные системы/os-intro$ git push
Enumerating objects: 38, done.
Counting objects: 100% (38/38), done.
Delta compression using up to 6 threads
Compressing objects: 100% (30/30), done.
Writing objects: 100% (37/37), 343.00 KiB | 2.17 MiB/s, done.
Total 37 (delta 4), reused 0 (delta 0)
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:kdzhofef/os-intro.git
36bd24d..8776bcc master -> master
kdzhofef@kdzhofef:~/work/study/2022-2023/Операционные системы/os-intro$
```

Рис. 2.10: Первый коммит

3 Вывод

Мы приобрели практические навыки работы с сервисом github.

4 Контрольные вопросы

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

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

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

- хранилище - пространство на накопителе где расположен репозиторий
- commit - сохранение состояния хранилища
- история - список изменений хранилища (коммитов)
- рабочая копия - локальная копия сетевого репозитория, в которой работает программист. Текущее состояние файлов проекта, основанное на версии, загруженной из хранилища (обычно на последней)

3. 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.

Распределенные системы контроля версий (Distributed Version Control System, DVCS) позволяют хранить репозиторий (его копию) у каждого разработчика, работающего с данной системой. При этом можно выделить центральный репозиторий (условно), в который будут отправляться изменения из локальных и, с ним же эти локальные репозитории будут синхронизироваться. При работе с такой системой, пользователи периодически синхронизируют свои локальные репозитории с центральным и работают непосредственно со своей локальной копией. После внесения достаточного количества изменений в локальную копию они (изменения) отправляются на сервер. При этом сервер, чаще всего, выбирается условно, т.к. в большинстве DVCS нет такого понятия как “выделенный сервер с центральным репозиторием”.

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

Один пользователь работает над проектом и по мере необходимости делает коммиты, сохраняя определенные этапы.

5. Опишите порядок работы с общим хранилищем VCS.

Несколько пользователей работают каждый над своей частью проекта. При этом каждый должен работать в своей ветки. При завершении работы ветка пользователя сливается с основной веткой проекта.

6. Каковы основные задачи, решаемые инструментальным средством git?

- Ведение истории версий проекта: журнал (log), метки (tags), ветвления (branches).

- Работа с изменениями: выявление (diff), слияние (patch, merge).
- Обеспечение совместной работы: получение версии с сервера, загрузка обновлений на сервер.

7. Назовите и дайте краткую характеристику командам git.

- git config - установка параметров
- git status - полный список изменений файлов, ожидающих коммита
- git add . - сделать все измененные файлы готовыми для коммита.
- git commit -m "[descriptive message]" - записать изменения с заданным сообщением.
- git branch - список всех локальных веток в текущей директории.
- git checkout [branch-name] - переключиться на указанную ветку и обновить рабочую директорию.
- git merge [branch] — соединить изменения в текущей ветке с изменениями из заданной.
- git push - запустить текущую ветку в удаленную ветку.
- git pull - загрузить историю и изменения удаленной ветки и произвести слияние с текущей веткой.

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

- git remote add [имя] [url] — добавляет удалённый репозиторий с заданным именем;
- git remote remove [имя] — удаляет удалённый репозиторий с заданным именем;
- git remote rename [старое имя] [новое имя] — переименовывает удалённый репозиторий;
- git remote set-url [имя] [url] — присваивает репозиторию с именем новый адрес;

- `git remote show [имя]` — показывает информацию о репозитории.

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

Ветвление — это возможность работать над разными версиями проекта: вместо одного списка с упорядоченными коммитами история будет расходиться в определённых точках. Каждая ветвь содержит легковесный указатель HEAD на последний коммит, что позволяет без лишних затрат создать много веток. Ветка по умолчанию называется `master`, но лучше назвать её в соответствии с разрабатываемой в ней функциональностью.

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

Зачастую нам не нужно, чтобы Git отслеживал все файлы в репозитории, потому что в их число могут входить:

Список литературы

1. Лекция Системы контроля версий
2. GitHub для начинающих