

Отчёт по лабораторной работе №2

Управление версиями

Скруха Евгений

Содержание

| | | |
|---|--------------------------------|----|
| 1 | Цель работы | 4 |
| 2 | Выполнение лабораторной работы | 5 |
| 3 | Вывод | 10 |
| 4 | Контрольные вопросы | 11 |

List of Figures

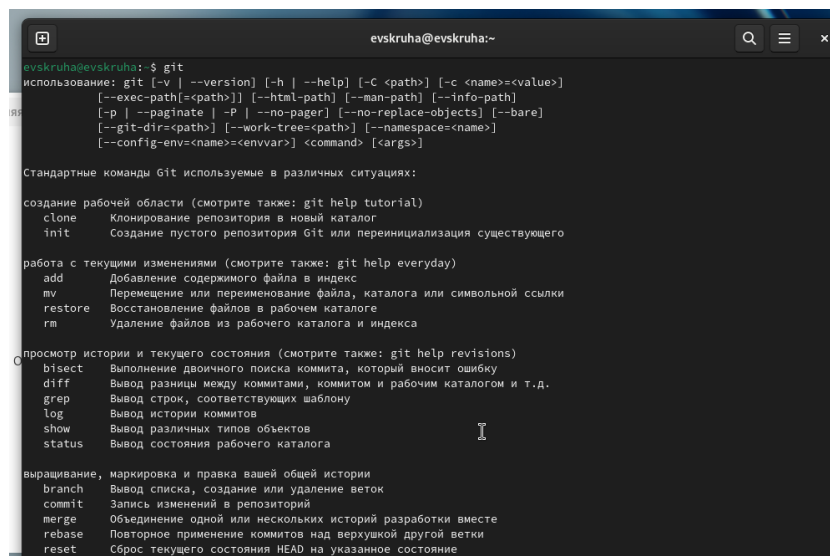
| | | |
|------|---|---|
| 2.1 | Загрузка пакетов | 5 |
| 2.2 | Параметры репозитория | 5 |
| 2.3 | rsa-4096 | 6 |
| 2.4 | ed25519 | 6 |
| 2.5 | GPG ключ | 7 |
| 2.6 | GPG ключ | 7 |
| 2.7 | Параметры репозитория | 8 |
| 2.8 | Связь репозитория с аккаунтом | 8 |
| 2.9 | Загрузка шаблона | 8 |
| 2.10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



```
evskruha@evskruha:~$ git
использование: git [-v | --version] [-h | --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>]
[--config-env=<name>=<envvar>] <command> [<args>]

Стандартные команды Git используются в различных ситуациях:

создание рабочей области (смотрите также: git help tutorial)
clone      Клонирование репозитория в новый каталог
init       Создание пустого репозитория Git или переинициализация существующего

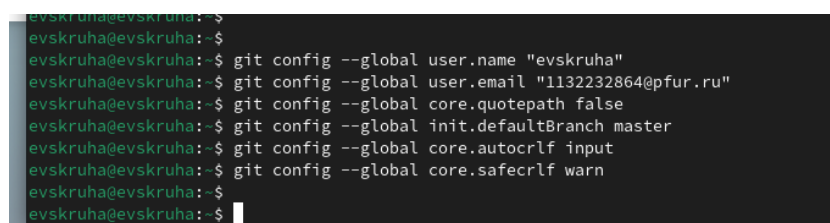
работа с текущими изменениями (смотрите также: git help everyday)
add        Добавление содержимого файла в индекс
mv         Перемещение или переименование файла, каталога или символической ссылки
restore    Восстановление файлов в рабочем каталоге
rm         Удаление файлов из рабочего каталога и индекса

просмотр истории и текущего состояния (смотрите также: git help revisions)
bisect     Выполнение двоичного поиска коммита, который вносит ошибку
diff       Вывод разницы между коммитами, коммитом и рабочим каталогом и т.д.
grep       Вывод строк, соответствующих шаблону
log        Вывод истории коммитов
show       Вывод различных типов объектов
status     Вывод состояния рабочего каталога

выращивание, маркировка и правка вашей общей истории
branch     Вывод списка, создание или удаление веток
commit     Запись изменений в репозиторий
merge      Объединение одной или нескольких историй разработки вместе
rebase     Повторное применение коммитов над вершущей другой ветки
reset      Сброс текущего состояния HEAD на указанное состояние
```

Figure 2.1: Загрузка пакетов

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



```
evskruha@evskruha:~$ git config --global user.name "evskruha"
evskruha@evskruha:~$ git config --global user.email "1132232864@pfur.ru"
evskruha@evskruha:~$ git config --global core.quotepath false
evskruha@evskruha:~$ git config --global init.defaultBranch master
evskruha@evskruha:~$ git config --global core.autocrlf input
evskruha@evskruha:~$ git config --global core.safecrlf warn
evskruha@evskruha:~$
```

Figure 2.2: Параметры репозитория

Создаем SSH ключи

```

evskruha@evskruha:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/evskruha/.ssh/id_rsa):
Created directory '/home/evskruha/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/evskruha/.ssh/id_rsa
Your public key has been saved in /home/evskruha/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Z1k5UKIt0YrHHIodz8qVNAUj4ZzHu1wDkAKCAP8+qCE evskruha@evskruha
The key's randomart image is:
+---[RSA 4096]-----+
|B .. +o+++.. |
|.o .oo+=.o . |
| . ++X*=. + |
| .. +.X+ o . |
| .. +S * |
| o o. = . |
|E . o o |
|.o . |
|. |
+-----[SHA256]-----+
evskruha@evskruha:~$

```

Figure 2.3: rsa-4096

```

evskruha@evskruha:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/evskruha/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/evskruha/.ssh/id_ed25519
Your public key has been saved in /home/evskruha/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:c652CK3Gb4UFLMmdtGQV7lxZPA5iXTejMEUJX8yvprw evskruha@evskruha
The key's randomart image is:
+---[ED25519 256]---+
| o.=+*B+=*o|
| =oo=++o*=|
| .o oo= o|
| .o . ..|
| .So. o . |
| . o+. o |
| . o o.. o |
| + +.. o |
| . +o. E. |
+-----[SHA256]-----+
evskruha@evskruha:~$

```

Figure 2.4: ed25519

Создаем GPG ключ

```
evskruha@evskruha:~  
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? (y/N) y  
GnuPG должен составить идентификатор пользователя для идентификации ключа.  
Ваше полное имя: evskruha  
Адрес электронной почты: 1132232864@pfur.ru  
Примечание:  
Вы выбрали следующий идентификатор пользователя:  
"evskruha <1132232864@pfur.ru>"  
Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход? O  
Необходимо получить много случайных чисел. Желательно, чтобы Вы  
в процессе генерации выполняли какие-то другие действия (печатать  
на клавиатуре, движения мыши, обращения к дискам); это даст генератору  
случайных чисел больше возможностей получить достаточное количество энтропии.  
Необходимо получить много случайных чисел. Желательно, чтобы Вы  
в процессе генерации выполняли какие-то другие действия (печатать  
на клавиатуре, движения мыши, обращения к дискам); это даст генератору  
случайных чисел больше возможностей получить достаточное количество энтропии.  
gpg: /home/evskruha/.gnupg/trustdb.gpg: создана таблица доверия  
gpg: создан каталог '/home/evskruha/.gnupg/openpgp-revocs.d'  
gpg: сертификат отзыва записан в '/home/evskruha/.gnupg/openpgp-revocs.d/22934FE115465FABE986D57CA1AFD843C1E0C7AC.rev'.  
открытый и секретный ключи созданы и подписаны.  
pub rsa4096 2024-06-19 [SC]  
22934FE115465FABE986D57CA1AFD843C1E0C7AC  
uid evskruha <1132232864@pfur.ru>  
sub rsa4096 2024-06-19 [E]  
evskruha@evskruha: $
```

Figure 2.5: GPG ключ

Добавляем GPG ключ в аккаунт

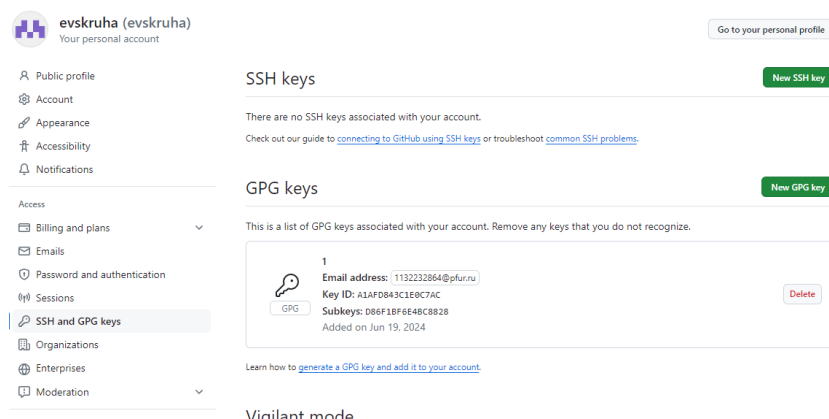


Figure 2.6: GPG ключ

Настройка автоматических подписей коммитов git

```
x6fPVBrs5jy5o1Tx3tuAZho0X8r2xLmF22uFcR6DL0Xp/fJrrkpjbwLRAmLVuk0d
B/0hjwM0jUndQXVNa3+949aRvkkQCBnWuVtIySm/au9CTm+pxf0/LmNVp7i3h5KY
UETWFGNYZTVjvhnoJ5uLLRLVWmNDntbtEIX9RZIYQCS2uqFsQzzrvBMZWpkGj7ep
SI0rEMACF5nf+3dpgBA3XTfVYCH6GT2JHpenu6JYqdv65r5xTzZ/xFnPcPAkDXM
MhstHjMcREZB7r9zJdCUhfru/1KshvysRe1pS3JLaS1p7fKL71JV6YgV0R91poeg
IIZQTsHLHLiwGqJew1x0yG5bsht7M3LJ6LHPC00c9Y10S98HCmuPIUwd8L4WjML
TTtqC5TNq7/lww4iN64C/PKrtZMwJYxd1Sr0ITpfKsX+lgLpfm4hAXnkQB26VC
zM9Eb6PdA+93tcseaI+HJ/XF44MNHekt6k0e/v2Y43lss5JG4gXor90xWhs=
=bkqv
-----END PGP PUBLIC KEY BLOCK-----
evskruha@evskruha:~$
evskruha@evskruha:~$
evskruha@evskruha:~$
evskruha@evskruha:~$ git config --global user.signingkey A1AFD843C1E0C7AC
evskruha@evskruha:~$ git config --global commit.gpgsign true
evskruha@evskruha:~$ git config --global gpg.program $(which gpg2)
evskruha@evskruha:~$
```

Figure 2.7: Параметры репозитория

Настройка gh

```
evskruha@evskruha:~$
evskruha@evskruha:~$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations on this host? SSH
? Upload your SSH public key to your GitHub account? /home/evskruha/.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: 3D65-1C1F
Press Enter to open github.com in your browser...
restorecon: SELinux: Could not get canonical path for /home/evskruha/.mozilla/firefox/*: restorecon: No such file or direc
tory.
✓ Authentication complete.
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
✓ Uploaded the SSH key to your GitHub account: /home/evskruha/.ssh/id_rsa.pub
✓ Logged in as evskruha
evskruha@evskruha:~$
```

Figure 2.8: Связь репозитория с аккаунтом

Загрузка шаблона репозитория и синхронизация

```
Клонирование в «/home/evskruha/work/study/2023-2024/Операционные системы/os-intro/template/presentation»...
remote: Enumerating objects: 95, done.
remote: Counting objects: 100% (95/95), done.
remote: Compressing objects: 100% (67/67), done.
remote: Total 95 (delta 34), reused 87 (delta 26), pack-reused 0
Получение объектов: 100% (95/95), 96.99 КБ | 1.04 МБ/с, готово.
Определение изменений: 100% (34/34), готово.
Клонирование в «/home/evskruha/work/study/2023-2024/Операционные системы/os-intro/template/report»...
remote: Enumerating objects: 126, done.
remote: Counting objects: 100% (126/126), done.
remote: Compressing objects: 100% (87/87), done.
remote: Total 126 (delta 52), reused 108 (delta 34), pack-reused 0
Получение объектов: 100% (126/126), 335.80 КБ | 1.96 МБ/с, готово.
Определение изменений: 100% (52/52), готово.
Submodule path 'template/presentation': checked out '40a1761813e197d00e8443ff1ca72c60a304f24c'
Submodule path 'template/report': checked out '7c31ab8e5dfa8cbb2d67caeb8a19ef8028ced88e'
evskruha@evskruha:~/work/study/2023-2024/Операционные системы/os-intro$ rm package.json
evskruha@evskruha:~/work/study/2023-2024/Операционные системы/os-intro$ make COURSE=os-intro prepare
CHANGELOG.md  COURSE  LICENSE  prepare  project-personal  README.git-flow.md  template
config        labs    Makefile  presentation  README.en.md  README.md
evskruha@evskruha:~/work/study/2023-2024/Операционные системы/os-intro$
```

Figure 2.9: Загрузка шаблона

Подготовка репозитория и коммит изменений


```

create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_fignos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_secnos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/core.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/main.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 project-personal/stage6/report/report.md
evskruha@evskruha: ~/work/study/2023-2024/Операционные системы/os-intro$ git push
Перечисление объектов: 38, готово.
Подсчет объектов: 100% (38/38), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (37/37), 342.06 КиБ | 2.78 МБ/с, готово.
Total 37 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:evskruha/os-intro.git
   5ec6285..fca4a15  master -> master
evskruha@evskruha: ~/work/study/2023-2024/Операционные системы/os-intro$
evskruha@evskruha: ~/work/study/2023-2024/Операционные системы/os-intro$
evskruha@evskruha: ~/work/study/2023-2024/Операционные системы/os-intro$

```

Figure 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: