

Лабораторная работа №2

Дисциплина: Операционные системы

Газизянов Владислав Альбертович

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Контрольные вопросы	10
3	Выводы	12
	Список литературы	13

Список иллюстраций

2.1	Git	6
2.2	Имя	6
2.3	Параметры	6
2.4	Ключи	7
2.5	Добавление ключей	8
2.6	Автоматические подписи	8
2.7	Авторизация	8
2.8	Удаление файлов и создание каталогов	9
2.9	Отправка	9

Список таблиц

1 Цель работы

Изучение работы и назначение системы контроля версий git приобретение навыков по работе с ней

2 Выполнение лабораторной работы

Устанавливаем git и fh



```
[sudo] пароль для vagazizyanov:
[root@vagazizyanov ~]# dnf install git
Fedora 39 - x86_64 - Updates      29 kB/s | 19 kB   00:00
Fedora 39 - x86_64 - Updates      2.0 MB/s | 2.5 MB 00:01
```

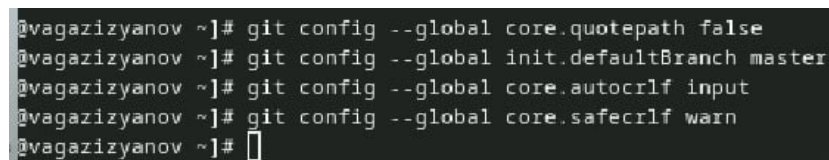
Рис. 2.1: Git

Задаём имя и email владельца репозитория, настраиваем utf-8 и параметры autocrlf and safecrlf



```
Введено!
@vagazizyanov ~]# git config --global user.name "Vladislav Gazizyan"
```

Рис. 2.2: Имя



```
@vagazizyanov ~]# git config --global core.quotePath false
@vagazizyanov ~]# git config --global init.defaultBranch master
@vagazizyanov ~]# git config --global core.autocrlf input
@vagazizyanov ~]# git config --global core.safecrlf warn
@vagazizyanov ~]#
```

Рис. 2.3: Параметры

Создаём ключ ssh и pgp

```

. . O + . . |
. + + . . O. |
* + .S .O.E |
% o . . . o |
* O .oo. . |
=.o O .+. |
.=O =. . . |
----[SHA256]-----+
root@vagazizyanov ~]# ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/root/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_ed25519
Your public key has been saved in /root/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:nhStG1SiUXZw5nmzVwtbKV6gMgzCXamyQRUSTz7cM6g root@vagazizyanov
The key's randomart image is:
--[ED25519 256]--+
. =*%+* . . |
E@.&.o . . . |
o. *.O = o = |
o. . . * + B . |
+. S . + . |
. = . . |
. o |
----[SHA256]-----+
root@vagazizyanov ~]# gpg --full-generate-key
gpg (GnuPG) 2.4.3; Copyright (C) 2023 g10 Code GmbH
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: создан каталог '/root/.gnupg'
Выберите тип ключа:
1) RSA and RSA

```

Рис. 2.4: Ключи

Авторизируемся на GitHub и добавим созданные ключи, скопировав их от-
печатки

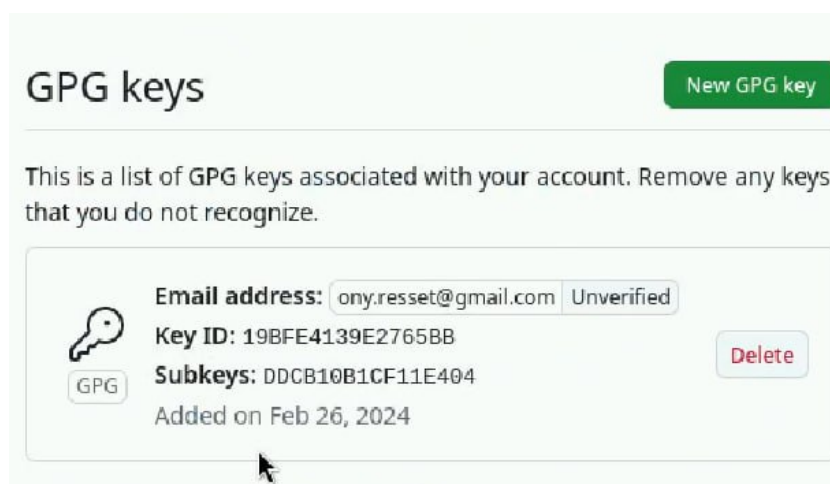


Рис. 2.5: Добавление ключей

Настройка автоматической подписи коммитов

```
zizyanov@vagazizyanov ~]$ git config --global user.signingkey ony.resset@gmail.com
zizyanov@vagazizyanov ~]$ git config --global commit.gpgsign true
zizyanov@vagazizyanov ~]$ git config --global gpg.program ${which gpg}
```

Рис. 2.6: Автоматические подписи

Авторизируемся с помощью команды `gh login auth` и, создав и перейдя в нужные каталоги создадим репозиторий на основе шаблона, а затем клонируем его себе

```
[vagazizyanov@vagazizyanov ~]$ 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
? Generate a new SSH key to add to your GitHub account? Yes
Enter a passphrase for your new SSH key (Optional)
File for your SSH key: GitHub CLI
Would you like to authenticate GitHub CLI? Login with a web browser
```

Рис. 2.7: Авторизация

Далее удаляем все лишние файлы и создаём необходимые каталоги


```
[vagazizyanov@vagazizyanov Операционные системы]$ cd ~/work/study/2022-2023/"Операционные системы"/os-intro
bash: cd: /home/vagazizyanov/work/study/2022-2023/Операционные системы/os-intro: Нет такого файла или каталога
[vagazizyanov@vagazizyanov Операционные системы]$ cd ~/work/study/2023-2024/"Операционные системы"/os-intro
[vagazizyanov@vagazizyanov os-intro]$ rm package.json
[vagazizyanov@vagazizyanov os-intro]$ echo os-intro > COURSE
[vagazizyanov@vagazizyanov os-intro]$ make
Usage:
  make <target>

Targets:
  -> List of courses
  -> Generate directories structure
  -> Update submodules
```

Рис. 2.8: Удаление файлов и создание каталогов

Отправляем все изменения на github

```
.jpg
create mode 100644 project-personal/stage6/report/pandoc/cs1/gost-r-7-0-5-8-numeric.cs1
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_eq
.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_fi
s.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_se
s.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_ta
nos.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxno
_init__.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxno
ore.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxno
ain.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxno
andocattributes.py
create mode 100644 project-personal/stage6/report/report.md
[vagazizyanov@vagazizyanov os-intro]$ git push
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
При сжатии изменений используется до 5 потоков
Сжать объектов: 100% (30/30), готово.
Сжать объектов: 100% (38/38), 342.11 КиБ | 2.50 МБ/с, готово.
38 (изменений 4), повторно использовано 0 (изменений 0), повторно исп
0 пакетов 0
e: Resolving deltas: 100% (4/4), completed with 1 local object.
github.com: ONYX8880/study_2023-2024_os-intro.git
930d9..cf31b5b master -> master
vagazizyanov@vagazizyanov os-intro]$
```

Рис. 2.9: Отправка

2.1 Контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются? Они применяются при работе нескольких человек с одним проектом. При внесении изменений позволяют фиксировать, совмещать и возвращать из-менения разных людей
2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия. Хранилище - место, где находятся данные(файлы, коды и тд) Commit - команда, для сохранения изменений История - информация о предыдущих изменениях Рабочая копия - одна из версий проекта, с которой ведется работа(= текущая/основная)
3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида. Централизованные системы предполагают наличие единого репозитория для хранения данных(CVC, Subversion) В децентрализованных системах центральный репозиторий не обязателен(Git, Bazaar)
4. Опишите действия с VCS при единоличной работе с хранилищем. абота происходит на своем компьютере, сначала обновляются данные, в конце они размещаются в центральном репозитории
5. Опишите порядок работы с общим хранилищем VCS. Для идентификации на сервере необходимы ключи и затем создание репозитория, только затем можно работать на локальной машине. Также в конце изменения добавляются на сервер
6. Каковы основные задачи, решаемые инструментальным средством git? хранение информации о всех изменениях обеспечение удобства командной работы
7. Назовите и дайте краткую характеристику командам git. 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 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 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 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 репозиториями. локальный репозиторий - работа со своими файлами удалённый репозиторий - совместная работа, общий проект
9. Что такое и зачем могут быть нужны ветви (branches)? Ветви это пути к отдельным состоящим (отделы проекта),они дают возможность вносить изменения только в часть проекта и не трогать все вышестоящее
10. Как и зачем можно игнорировать некоторые файлы при commit? Игнорировать файлы можно с помощью `.gitignore`. Нужно это, например, при наличии ненужных(лишних или созданных автоматически) файлов.

3 Выводы

В ходе работы была освоена работа с системой контроля версий, был установлен git, проведена авторизация, заданы базовые настройки, созданы ключи, клонировался репозиторий и так далее.

Список литературы