

Отчет по лабораторной работе №2

Первоначальна настройка git

Акопян Сатеник Манвеловна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	14
	Список литературы	15

Список иллюстраций

2.1	рисунок 1	6
2.2	рисунок 2	6
2.3	рисунок 3	6
2.4	рисунок 4	7
2.5	рисунок 5	7
2.6	рисунок 6	7
2.7	рисунок 7	8
2.8	рисунок 8	8
2.9	рисунок 9	8
2.10	рисунок 10	8
2.11	рисунок 11	9
2.12	рисунок 12	9

Список таблиц

1 Цель работы

Изучить идеологию и применение средств контроля версий.

Освоить умения по работе с git.

2 Выполнение лабораторной работы

1. Установка программного обеспечения

```
[smakopyan@fedora report]$ git --version
git version 2.39.1
[smakopyan@fedora report]$ gh --version
gh version 2.22.1 (2023-01-29)
https://github.com/cli/cli/releases/tag/v2.22.1
[smakopyan@fedora report]$
```

Рис. 2.1: рисунок 1

2. Базовая настройка git. Сделаем предварительную конфигурацию git, а так же настроим utf-8 в выводе сообщений git и зададим имя начальной ветки (будем называть её master)

```
smakopyan@fedora:~
[smakopyan@fedora ~]$ git config --global user.name "<smakopyan>"
[smakopyan@fedora ~]$ git config --global user.email "<satenikak@yandex.ru>"
[smakopyan@fedora ~]$ git config --global core.quotePath false
[smakopyan@fedora ~]$ git config --global init.defaultBranch master
[smakopyan@fedora ~]$ git config --global core.autocrlf input
[smakopyan@fedora ~]$ git config --global core.safecrlf warn
[smakopyan@fedora ~]$
```

Рис. 2.2: рисунок 2

3. Создание ключей ssh.

```
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
```

Рис. 2.3: рисунок 3

```
+----[SHA256]-----+
[root@fedora ~]# ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/root/.ssh/id_ed25519):
```

Рис. 2.4: рисунок 4

4.Создание ключей pgr

```
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'
gpg: создан щит с ключами '/root/.gnupg/pubring.kbx'
Выберите тип ключа:
(1) RSA and RSA
(2) DSA and Elgamal
(3) DSA (sign only)
(4) RSA (sign only)
(9) ECC (sign and encrypt) *default*
(10) ECC (только для подписи)
(14) Existing key from card
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
```

Рис. 2.5: рисунок 5

5.Настройка github

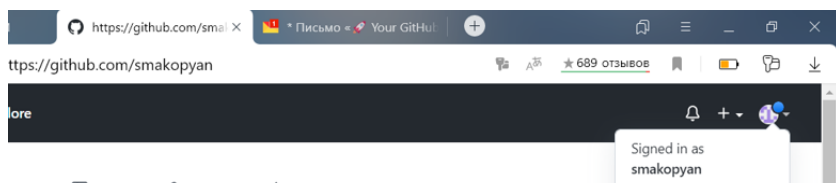


Рис. 2.6: рисунок 6

6.Добавление PGP ключа в GitHub

```
[root@fedora ~]# gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: глубина: 0 достоверных: 1 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f, 1u
/root/.gnupg/pubring.kbx
-----
sec   rsa4096/1283D934F4AE398D 2023-02-16 [SC]
      EB0963A439A73A4B9BC2E4391283D934F4AE398D
uid           [ абсолютно ] Satenik <satenikak@yandex.ru>
ssb   rsa4096/6C56157D3630577D 2023-02-16 [E]

[root@fedora ~]#
```

Рис. 2.7: рисунок 7

```
-bash: синтаксическая ошибка рядом с неожиданным маркером «|»
[root@fedora ~]# gpg --armor --export 1283D934F4AE398D | xclip -sel clip
[root@fedora ~]#
```

Рис. 2.8: рисунок 8

7. Настройка автоматических подписей коммитов git

```
[root@fedora ~]# git config --global user.signinkey 1283D934F4AE398D
[root@fedora ~]# git config --global commit.gpgsign true
[root@fedora ~]# git config --global gpg.program $(which gpg2)
[root@fedora ~]#
```

Рис. 2.9: рисунок 9

8. Настройка gh

```
[root@fedora ~]# 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? Skip
? How would you like to authenticate GitHub CLI? Paste an authentication token
Tip: you can generate a Personal Access Token here https://github.com/settings/tokens
The minimum required scopes are 'repo', 'read:org'.
? Paste your authentication token: *****
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
✓ Logged in as smakopyan
[root@fedora ~]#
```

Рис. 2.10: рисунок 10

9. Создание репозитория курса на основе шаблона


```
[smakopyan@fedora Операционные системы]$ mkdir -p ~/work/study/2022-2023/"Операционные системы"
[smakopyan@fedora Операционные системы]$ cd ~/work/study/2022-2023/"Операционные системы"
[smakopyan@fedora Операционные системы]$ gh repo create study_2022-2023_os-intro --template=yamadharm/course-directory-student-template --public
✓ Created repository smakopyan/study_2022-2023_os-intro on GitHub
[smakopyan@fedora Операционные системы]$ git clone --recursive git@github.com:smakopyan/study_2022-2023_os-intro.git os-intro
bash: owner: Нет такого файла или каталога
[smakopyan@fedora Операционные системы]$ git clone --recursive git@github.com:smakopyan/study_2022-2023_os-intro.git os-intro
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 | 8.47 Миб/с, готово.
Определение изменений: 100% (1/1), готово.
```

Рис. 2.11: рисунок 11

```
[smakopyan@fedora Операционные системы]$ cd ~/work/study/2022-2023/"Операционные системы"/os-intro
[smakopyan@fedora os-intro]$ rm package.json
[smakopyan@fedora os-intro]$ echo os-intro > COURSE
[smakopyan@fedora os-intro]$ make
[smakopyan@fedora os-intro]$
```

Рис. 2.12: рисунок 12

```
[smakopyan@fedora os-intro]$ git add .
[smakopyan@fedora os-intro]$ git commit -am 'a'
[master 4068b12] a
361 files changed, 100327 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 labs/lab01/presentation/Makefile
create mode 100644 labs/lab01/presentation/image/kulyabov.jpg
create mode 100644 labs/lab01/presentation/presentation.md
create mode 100644 labs/lab01/report/Makefile
create mode 100644 labs/lab01/report/bib/cite.bib
```

Контрольные вопро-

сы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначены? Системы контроля версий (Version Control System, VCS) применяются при работе с несколькими версиями документов.

2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочий репозиторий. Репозиторий - хранилище версий - в нем хранятся все документы вместе с историей изменений.

изменения и другой служебной информацией

commit - редактирует репозиторий, заносит туда изменения

Рабочая копия - копия проекта, связанная с репозиторием

3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS

Централизованные VCS : одно основное хранилище всего проекта, каждый пользователь

Децентрализованные VCS: У каждого пользователя свой вариант (возможно не

один) репозитория, присутствует возможность добавлять и забирать изменения из любого репозитория

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

Сначала создаем и подключаем удаленный репозиторий, затем по мере изменения проек

5. Опишите порядок работы с общим хранилищем VCS.

Пользователь перед началом работы получает нужную ему версию файлов, после внесен

6. Каковы основные задачи, решаемые инструментальным средством git?

Основными задачами, решаемыми инструментальным средством git являются: хранение и

7. 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 рабочей области):

```
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
```

(при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана)

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

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

Ветви нужны для совместной работы над проектом.

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

Т.к. во время работы над проектом могут создаваться файлы, которые не должны попа

3 Выводы

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

Список литературы