

Отчёт по лабораторной работе №3 (отчёт по лабораторной работе №2)

Markdown

Аделина Руслановна Галиева

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Контрольные вопросы	12
4	Выводы	16

Список иллюстраций

2.1	1.png	6
2.2	2.png	6
2.3	3.png	6
2.4	4.png	6
2.5	5.png	7
2.6	6.png	7
2.7	7.png	7
2.8	8.png	8
2.9	9.png	8
2.10	10.png	9
2.11	11.png	9
2.12	12.png	9
2.13	13.png	9
2.14	14.png	10
2.15	15.png	10
2.16	16.png	10
2.17	17.png	10
2.18	18.png	11

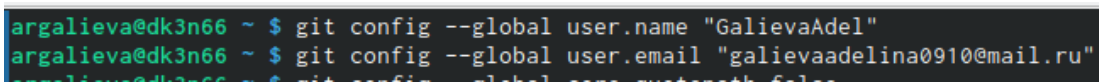
Список таблиц

1 Цель работы

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

2 Выполнение лабораторной работы

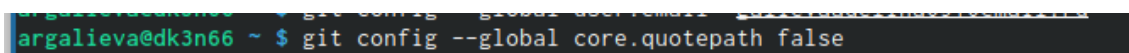
1. Зададим имя и email владельца репозитория (рис. 2.1)



```
argalievadk3n66 ~ $ git config --global user.name "GalievaAdel"
argalievadk3n66 ~ $ git config --global user.email "galievaadelina0910@mail.ru"
```

Рис. 2.1: 1.png

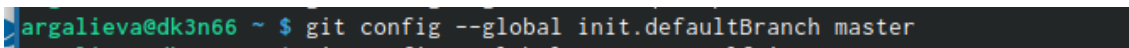
2. Настроим utf-8 в выводе сообщений git (рис. 2.2)



```
argalievadk3n66 ~ $ git config --global core.quotePath false
```

Рис. 2.2: 2.png

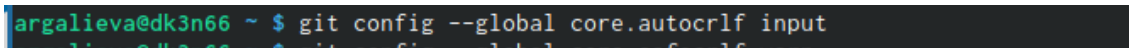
3. Зададим имя начальной ветки (рис. 2.3)



```
argalievadk3n66 ~ $ git config --global init.defaultBranch master
```

Рис. 2.3: 3.png

4. Параметр autocrlf (рис. 2.4)



```
argalievadk3n66 ~ $ git config --global core.autocrlf input
```

Рис. 2.4: 4.png

5. Параметр safecrLf (рис. 2.5)

```
argalievadk3n66 ~ $ git config --global core.safecrLf warn
```

Рис. 2.5: 5.png

6. По алгоритму rsa с ключём размером 4096 бит (рис. 2.6)

```
argalievadk3n66 ~ $ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/afs/.dk.sci.pfu.edu.ru/home/a/r/argalievadk3n66/.ssh/id_rsa):
/afs/.dk.sci.pfu.edu.ru/home/a/r/argalievadk3n66/.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 /afs/.dk.sci.pfu.edu.ru/home/a/r/argalievadk3n66/.ssh/id_rsa
Your public key has been saved in /afs/.dk.sci.pfu.edu.ru/home/a/r/argalievadk3n66/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:s7Pp/PfBLLb68Xm0whUsUGXHfwNMwfMaBmjc02QCHGk argalievadk3n66
The key's randomart image is:
+---[RSA 4096]-----+
|          .o= o.=++..|
|          E =. * ..|
```

Рис. 2.6: 6.png

7. По алгоритму ed25519 (рис. 2.7)

```
argalievadk3n66 ~ $ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/afs/.dk.sci.pfu.edu.ru/home/a/r/argalievadk3n66/.ssh/id_ed25519):
/afs/.dk.sci.pfu.edu.ru/home/a/r/argalievadk3n66/.ssh/id_ed25519 already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /afs/.dk.sci.pfu.edu.ru/home/a/r/argalievadk3n66/.ssh/id_ed25519
Your public key has been saved in /afs/.dk.sci.pfu.edu.ru/home/a/r/argalievadk3n66/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:9Ci5TWN0VsaGb3J2a0cYbmsIVz88TwhwK+Egw58osg argalievadk3n66
The key's randomart image is:
+--[ED25519 256]--+
|      .+...  |
|      . o= * .|
```

Рис. 2.7: 7.png

8. Генерируем ключ (рис. 2.8)

```

argalieva@dk3n66 ~ $ gpg --full-generate-key
gpg (GnuPG) 2.2.40; Copyright (C) 2022 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.

Выберите тип ключа:
  (1) RSA и RSA (по умолчанию)
  (2) DSA и Elgamal
  (3) DSA (только для подписи)
  (4) RSA (только для подписи)
  (14) Имеющийся на карте ключ
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

```

Рис. 2.8: 8.png

9. Выводим список ключей и копируем отпечаток приватного ключа (рис. 2.9)

```

argalieva@dk3n66 ~ $ 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: 3 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f, 3u
/afs/.dk.sci.pfu.edu.ru/home/a/r/argalieva/.gnupg/pubring.kbx
-----
sec  rsa4096/66BE96C0F6D255BD 2023-02-18 [SC]
      53027A13C9018BCEC34D43A166BE96C0F6D255BD
uid          [ абсолютно ] GalievaAdel <galievaadelina0910@mail.ru>
ssb  rsa4096/63211877CBD21A9A 2023-02-18 [E]

sec  rsa4096/9D4E14F3E2CD4068 2023-02-18 [SC]
      7713CCF21A46E0A4FECF7F4F9D4E14F3E2CD4068
uid          [ абсолютно ] GalievaAdel <galievaadelina0910@mail.ru>
ssb  rsa4096/CBC387F93EF1A44F 2023-02-18 [E]

sec  rsa4096/9617A75191AF3773 2023-02-18 [SC]
      FD6BCF0F8844FD2ACAE175649617A75191AF3773
uid          [ абсолютно ] GalievaAdel <galievaadelina0910@mail.ru>
ssb  rsa4096/831358926067B6DA 2023-02-18 [E]

```

Рис. 2.9: 9.png

10. Копируем наш сгенерированный PGP ключ в буфер обмена (рис. 2.10)


```
argalievadk3n66 ~ $ gpg --armor --export 66BE96C0F6D255BD | xclip -sel clip
```

Рис. 2.10: 10.png

11. Используя введённый email, указываем Git применять его при подписи коммитов (рис. 2.11)

```
argalievadk3n66 ~ $ git config --global user.signingkey 66BE96C0F6D255BD
argalievadk3n66 ~ $ git config --global commit.gpgsign true
argalievadk3n66 ~ $ git config --global gpg.program $(which gpg2)
```

Рис. 2.11: 11.png

12. Авторизовываемся (рис. 2.12)

```
argalievadk3n66 ~ $ gh auth login
? What account do you want to log into? GitHub.com
? You're already logged into github.com. Do you want to re-authenticate? Yes
? What is your preferred protocol for Git operations? HTTPS
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 3852-BBCF
Press Enter to open github.com in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol https
✓ Configured git protocol
✓ Logged in as GalievaAdel
```

Рис. 2.12: 12.png

13. Создаём репозиторий (рис. 2.13)

```
argalievadk3n66 ~ $ mkdir -p ~/work/study/2022-2023/"Операционные системы"
argalievadk3n66 ~ $ cd ~/work/study/2022-2023/"Операционные системы"
argalievadk3n66 ~/work/study/2022-2023/Операционные системы $ gh repo create study_2022-2023_os-intro --template=yamadharma/course-directory-student-template --public
✓ Created repository GalievaAdel/study_2022-2023_os-intro on GitHub
argalievadk3n66 ~/work/study/2022-2023/Операционные системы $ git clone --recursive https://github.com/GalievaAdel/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
```

Рис. 2.13: 13.png

14. Переходим в каталог курса (рис. 2.14)

```
argalievadk3n66 ~/work/study/2022-2023/Операционные системы $ cd ~/work/study/2022-2023/Операционные системы/os-intro
```

Рис. 2.14: 14.png

15. Удаляем лишние файлы (рис. 2.15)

```
argalievadk3n66 ~/work/study/2022-2023/Операционные системы/os-intro $ rm package.json
```

Рис. 2.15: 15.png

16. Создаём необходимые каталоги (рис. 2.16)

```
argalievadk3n66 ~/work/study/2022-2023/Операционные системы/os-intro $ echo os-intro > COURSE
argalievadk3n66 ~/work/study/2022-2023/Операционные системы/os-intro $ make
```

Рис. 2.16: 16.png

17. Отправляем файлы на сервер (рис. 2.17) (рис. 2.18)

```
argalievadk3n66 ~/work/study/2022-2023/Операционные системы/os-intro $ git add .
argalievadk3n66 ~/work/study/2022-2023/Операционные системы/os-intro $ git commit -am 'feat(main): make course structure'
[master e924e4d] feat(main): make course structure
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
create mode 100644 labs/lab01/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab01/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_eqnos.py
```

Рис. 2.17: 17.png

```
argalievadk3n66 ~/work/study/2022-2023/Операционные системы/os-intro $ git push
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (38/38), 343.05 КиБ | 14.92 МиБ/с, готово.
Всего 38 (изменений 4), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To https://github.com/GalievaAdel/os-intro.git
 88cfdd8..e924e4d master -> master
argalievadk3n66 ~/work/study/2022-2023/Операционные системы/os-intro $
```

Рис. 2.18: 18.png

3 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

4 Выводы

Я изучила идеологию и применение средств контроля версий. Освоила умения по работе с git.