

Лабораторная работа №2

Первоначальная настройка git

Дисциплина Операционные системы

Осокин Георгий Иванович НММбд-02-22

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Настройка GIT	6
2.2	Создание рабочего пространства	8
3	Ответы на контрольные вопросы	11
4	Выводы	14

Список иллюстраций

2.1	Конфигурация GIT	6
2.2	Создание PGP ключа	7
2.3	Подпись коммитов PGP ключем	7
2.4	Добавление ключа на гитхаб	7
2.5	Авторизация на гитхабе	8
2.6	Создание шаблона из шаблона	9
2.7	Клонирование репозитория	9
2.8	Удаление файла package.json и создание файла COURSE	9
2.9	Git push	10
2.10	Просмотр нового коммита на гитхабе	10

Список таблиц

1 Цель работы

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

2 Выполнение лабораторной работы

2.1 Настройка GIT

Сконфигурируем git и создадим SSH ключ

```
[giosokin@localhost ~]$ git config --global user.name "giosokin"
[giosokin@localhost ~]$ git config --global user.email "113226517Qpfur.ru"
[giosokin@localhost ~]$ git config --global core.quotepath false
[giosokin@localhost ~]$ git config --global core.autocrlf input
[giosokin@localhost ~]$ git config --global core.safecrlf warn
[giosokin@localhost ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/giosokin/.ssh/id_rsa): lab_key
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in lab_key
Your public key has been saved in lab_key.pub
The key fingerprint is:
SHA256:We9KM4VhiH979taWCQXHLRboPLrIG7PkSiPGLumgfJQ giosokin@teapod
The key's randomart image is:
+---[RSA 4096]-----+
|          .o..|
|         . . oo.|
|        . . +o .o.|
|       . + ++  .|
|      . S o.o..|
|     E   .+. .|
|    . .+ o.+.=.+ ..o|
|   o.o+ o +o=.* ..oo|
|  .oo.. .+. . . .|
+---[SHA256]-----+
[giosokin@localhost ~]$
```

Рис. 2.1: Конфигурация GIT

Создадим PGP ключ

```
[giosokin@localhost ~]$ 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.

gpg: directory '/home/giosokin/.gnupg' created
gpg: keybox '/home/giosokin/.gnupg/pubring.kbx' created
Please select what kind of key you want:
  (1) RSA and RSA (default)
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysizes do you want? (3072) 4096
Requested keysizes is 4096 bits
Please specify how long the key should be valid.
  0 = key does not expire
  <n> = key expires in n days
  <n>w = key expires in n weeks
  <n>m = key expires in n months
  <n>y = key expires in n years
Key is valid for? (0) 0
Key does not expire at all
Is this correct? (y/N) y
```

Рис. 2.2: Создание PGP ключа

Настроим подпись коммитов нашим PGP ключом

```
[giosokin@localhost ~]$ git config --global user.signingkey 4D5A3602EF8DE117
[giosokin@localhost ~]$ git config --global commit.gpgsign true
[giosokin@localhost ~]$ git config --global gpg.program $(which gpg2)
[giosokin@localhost ~]$ echo $(which gpg2)
/sbin/gpg2
```

Рис. 2.3: Подпись коммитов PGP ключем

Добавим PGP ключ на гитхаб.

С помощью команды `gpg --armor --export <PGP Fingerprint> | xclip -sel clip` скопируем PGP ключ И вставим в гитхаб

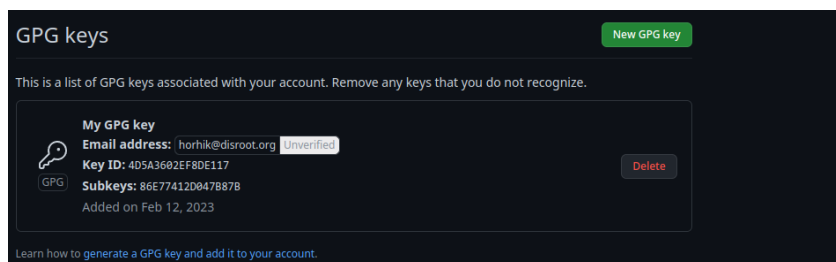


Рис. 2.4: Добавление ключа на гитхаб

Авторизируемся на гитхабе с помощью команды `gh`

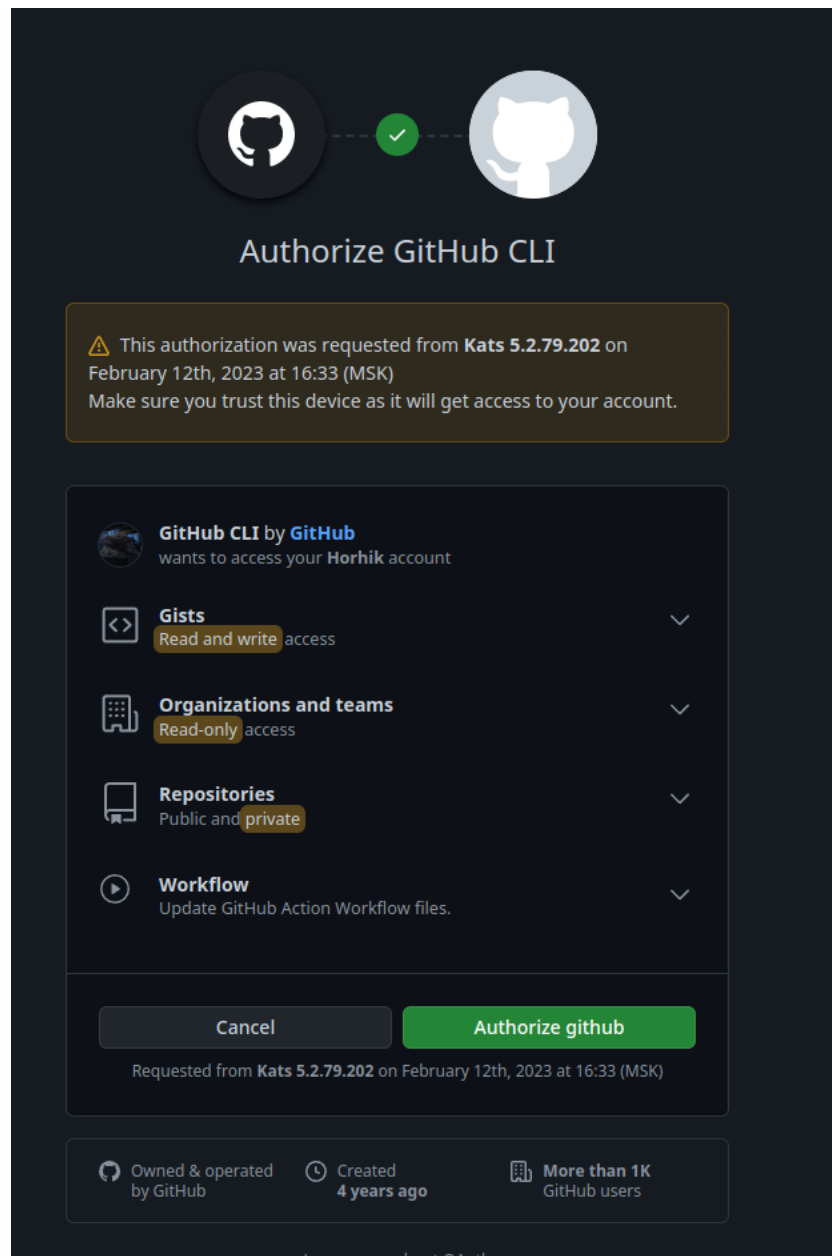


Рис. 2.5: Авторизация на гитхабе

2.2 Создание рабочего пространства

Перейдем в репозиторий с шаблоном и создадим из него шаблон

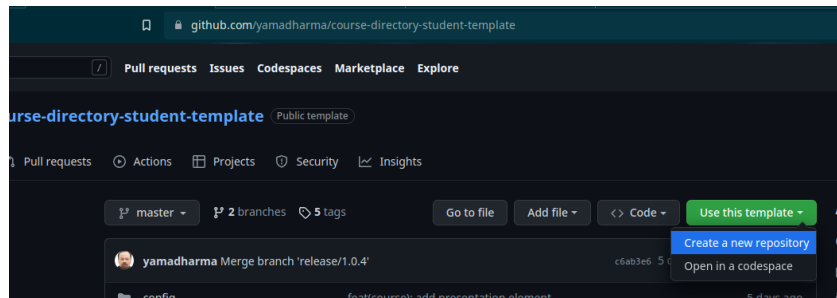


Рис. 2.6: Создание шаблона из шаблона

После создания шаблона, склонируем репозиторий на локальную машину

```
[giosokin@localhost Операционные системы]$ git clone --recursive git@github.com:horhik/study_2022-2023_os-intro
Cloning into 'study_2022-2023_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
Receiving objects: 100% (27/27), 16.92 KiB | 16.92 MiB/s, done.
Resolving deltas: 100% (1/1), done.
Submodule 'template/presentation' (https://github.com/yamadharma/academic-presentation-markdown-template.git) registered for path 'template/presentation'
Submodule 'template/report' (https://github.com/yamadharma/academic-laboratory-report-template.git) registered for path 'template/report'
Cloning into '/home/giosokin/work/study/2022-2023/Операционные системы/study_2022-2023_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
Receiving objects: 100% (82/82), 92.90 KiB | 669.60 KiB/s, done.
Resolving deltas: 100% (28/28), done.
Cloning into '/home/giosokin/work/study/2022-2023/Операционные системы/study_2022-2023_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 | 1.14 MiB/s, done.
Resolving deltas: 100% (40/40), done.
Submodule path 'template/presentation': checked out 'b1be3800ee91f5809264cb755d316174540b753e'
Submodule path 'template/report': checked out '1d1b61dcac9c287a83917b82e3aef11a33b1e3b2'
[giosokin@localhost Операционные системы]$ ls
study_2022-2023_os-intro
[giosokin@localhost Операционные системы]$
```

Рис. 2.7: Клонирование репозитория

Перейдем в репозиторий и удалим package.json Также создадим файл COURSE с текстом “os-intro”

```
[giosokin@localhost Операционные системы]$ cd study_2022-2023_os-intro/
[giosokin@localhost study_2022-2023_os-intro]$ ls
CHANGELOG.md  config  COURSE  LICENSE  Makefile  package.json  README.en.md  README.git-flow.md  README.md  template
[giosokin@localhost study_2022-2023_os-intro]$ rm package.json
[giosokin@localhost study_2022-2023_os-intro]$ echo os-intro > COURSE
[giosokin@localhost study_2022-2023_os-intro]$ make
[giosokin@localhost study_2022-2023_os-intro]$ ls
CHANGELOG.md  COURSE  LICENSE  prepare  project-personal  README.git-flow.md  template
config  Makefile  presentation  README.en.md  README.md
[giosokin@localhost study_2022-2023_os-intro]$
```

Рис. 2.8: Удаление файла package.json и создание файла COURSE

также запустим команду make, которая сгенерирует файловую структуру. Убедимся командой ls

Теперь запустим на гитхаб изменения. До этого исполнив `git add` и `git commit` с названием коммита `feat(main): make course structure`

```
create mode 100644 project-personal/stage6/report/report.md
[giosokin@localhost study_2022-2023_os-intro]$ git push
Enumerating objects: 40, done.
Counting objects: 100% (40/40), done.
Delta compression using up to 8 threads
Compressing objects: 100% (30/30), done.
Writing objects: 100% (38/38), 343.04 KiB | 2.03 MiB/s, done.
Total 38 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
remote: This repository moved. Please use the new location:
remote:   git@github.com:Horhik/study_2022-2023_os-intro.git
To github.com:horhik/study_2022-2023_os-intro
   951fc25..9153bd4  master -> master
[giosokin@localhost study_2022-2023_os-intro]$
```

Рис. 2.9: Git push

Зайдем на гитхаб и убедимся что изменения запустились

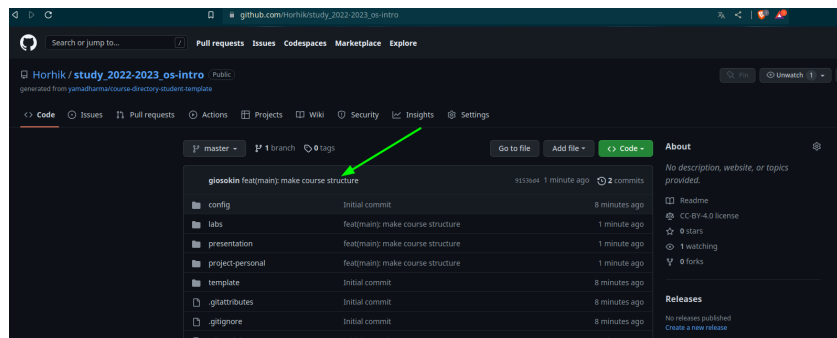


Рис. 2.10: Просмотр нового коммита на гихабе

3 Ответы на контрольные вопросы

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

Системы контроля версий применяются для фиксации изменений в программе и коллективной разработки

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

Хранилище: хранилище всех изменений и версий Commit: Единица изменения
История: Последовательность коммитов Рабочая копия: Последняя версия

3.0.0.3 Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.

Децентрализованные VCS скачивают на хост всю цепочку изменений, а централизованные позволяют только подключаться к полноценному хранилищу и запрашивать конкретную версию.

3.0.0.4 Опишите действия с VCS при единоличной работе с хранилищем.

Пользователь может клонировать версию на свою машину, внести изменения и создать ветку

3.0.0.5 Опишите порядок работы с общим хранилищем VCS.

Пользователь запрашивает у хранилища определенную версию программы, VCS выдает ему эту версию, он вносит туда изменения и отправляет их обратно. VCS сохраняет изменения новой версии

3.0.0.6 Каковы основные задачи, решаемые инструментальным средством git?

1. Разработка больших проектов командами
2. Контроль изменений множества файлов и откат изменений
3. Возможность сохранять несколько состояний разработки программы

3.0.0.7 Назовите и дайте краткую характеристику командам git.

`git add` - добавить изменения в коммит

`git commit` - сохранить изменения присвоив им название

`git push` - отправить изменения на удаленный репозиторий

`git config` - изменение параметров git

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

В большом проекте разработчики будут клонировать локальный репозиторий себе на устройства, благодаря чему смогут работать параллельно.

Если же репозиторий слишком большой, разработчики могут подключаться к нему удаленно и править код в новых ветках и т.п.

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

Ветви могут быть нужны для совместной работы над проектом, для разработки нескольких функций программы одновременно, что бы не конфликтовать файлами при параллельной разработки программы несколькими людьми

3.0.0.10 Как и зачем можно игнорировать некоторые файлы при commit?

Файлы стоит вносить в .gitignore, это сделано для того, что бы не загружать репозиторий лишними файлами или избегать утечки личной информации (баз данных и так далее)

4 Выводы

Мы изучили идеологию применения средств контроля версий и освоили базовые команды утилиты git.