

Лабораторная работа № 1 по дисциплине Математическое моделирование.

Миша Нкого Хосе Адольфо Мба НФИбд-02-19

10 февраля, 2022, Москва, Россия

Российский Университет Дружбы Народов

Цели и задачи работы

Цель лабораторной работы

Целью данной работы мы ставим изучение и применения средств контроля версий git, а также работу с различными командами в консоли.

Задачи лабораторной работы


1. Создать учетную запись на github.com
2. Создать и настроить репозиторий
3. Изучить механизм управления версиями
4. Изучить команды для работы с ветками

Ход выполнения работы


Создаем учетную запись и репозиторий

Owner *

Repository name *


 timbow64

 /


MatMod 

Great repository names are short and memorable. Need inspiration? How about [bug-free-octo-system?](#)

Description (optional)

☒  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☐ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**

Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**

A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

Figure 1: Создание репозитория

Инициализируем данный локальный репозиторий

```
PS C:\work> git init  
Initialized empty Git repository in C:/work/.git/  
PS C:\work> echo "# лабораторные работы" >> README.md  
PS C:\work> git add README.md  
PS C:\work>
```

Figure 2: Инициализация репозитория

Создаем SSH-ключ. Прописываем его в настройках на сайте.

```
PS C:\work> git config --global user.name timbow64
PS C:\work> git config --global user.email "1032189237@pfur.ru"
PS C:\work> git commit -m "first commit"
[master (root-commit) 41c0f33] first commit
1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 README.md
PS C:\work> ssh-keygen -C "timbow64 1032189063@pfur.ru"
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\User/.ssh/id_rsa):
Created directory 'C:\Users\User/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\User/.ssh/id_rsa.
Your public key has been saved in C:\Users\User/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:3/YLCIQ51ke1cTXwIyw2OZ0ANod65m84Nu/Jz7Ppfd4 timbow64 1032189063@pfur.
ru
The key's randomart image is:
+---[RSA 2048]-----+
  oo.o ooo
  ==.. B . .
  o*oo X o o
  ..o+ . = . .
  + S
  . o o
  o o +
  =_o.oEo
  *oB=. .o.
+---[SHA256]-----+
PS C:\work> cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCeJEXrZusmXkz1VqFMgO87oHwYwynJR5IHgbzw
XAGVvgn9xhtZmuACTX+4+Kpw2XsDYs118SLPPzVgVgkinDe+AYDfrc1sXmveKd1rcm7fbdc0fkou
TpmAv9Pya2Gpdq+d3r7te91sgAml6g08TyKx/zjXYTHdfJJzWswLAU1Zmfq8t8Be+fnfBVHvgVUN
HLH0nu29wsHqj1+WEd3se7W/v/51Cb/hn5VhN/8P36GchbNKvKyeXZ1hyy/IX08TwrG2XYW7kmfV
bht7cnkDI479c5Fzvoop+XBFY11+A3CZ1csIEGLThpuSOT4WANTAA3K/kmo/fd+SBXV8jS1Haq1fZ
timbow64 1032189063@pfur.ru
PS C:\work>
```

Figure 3: Создание SSH-ключа

Добавление ключа на github.com

SSH keys / Add new

Title

key

Key

```
ssh-rsa
AAAAAB3NzaC1yc2EAAAADAQABAAQCEjEXrZusmXkz1VqFMgO87oHwYwynJR5IHgbzwXAGVvgN9xhIZmuAcIX+4+Kp
W2XsDYsi18SLPPzVgVgKinDe+AYDfrCIsxMvEKd1rCm7fbc0FkQUtpimAv9PyA2Gpdq+d3r7te91sgAmL6g08TyKx/zjXYTH
dfIJzwSwLAU1ZmFq8t8bE+fnf8HVgVUNHLLHQnuz9wsHqJI+WEd3sE7W/v7SICb/hN5VhN/8P36GchbNKvKyeXZ1hyy/IXOB
TwRg2XYW7kmfVbht7cnKDI479C5FZVOo+XBFY11+A3CZicsIEGLThpuSOT4WANTaa3K/kmo/fd+SBXV8jS1HaqjFZ
timbow64 1032189063@pfur.ru
```

Add SSH key

Figure 4: Добавление ключа на github.com

Загружаем и отправляем файлы лицензионного соглашения и gitignore в сетевой репозиторий.

```
PS C:\work> git remote add origin git@github.com:timbow64/MatMod.git
PS C:\work> wget https://creativecommons.org/licenses/by/4.0/legalcode.txt -
O LICENSE
PS C:\work> wget https://www.toptal.com/developers/gitignore/api/python -O .
gitignore
PS C:\work> git add .
warning: LF will be replaced by CRLF in .gitignore.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in LICENSE.
The file will have its original line endings in your working directory
PS C:\work> git commit -am "add license"
[master 1d92a72] add license
2 files changed, 555 insertions(+)
create mode 100644 .gitignore
create mode 100644 LICENSE
PS C:\work> git push -u origin master
The authenticity of host 'github.com (140.82.121.3)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOq
U.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (7/7), 7.70 KiB | 2.57 MiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:timbow64/MatMod.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
PS C:\work> git push
Everything up-to-date
```

Figure 5: Загрузка файлов лицензии

Изучаем систему управления версиями файлов. Создаем ветку, начинаем и завершаем в ней релиз.

```
PS C:\works> git flow init
Which branch should be used for bringing forth production releases?
- master
Branch name for production releases: [master]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? [v]
Hooks and filters directory? [c:/work/.git/hooks]
PS C:\works> git branch
* master
PS C:\works> git flow release start 1.0.0
Switched to a new branch "release/1.0.0"

Summary of actions:
- A new branch 'release/1.0.0' was created, based on 'develop'
- You are now on branch 'release/1.0.0'

Follow-up actions:
- Bump the version number now!
- Start committing last-minute fixes in preparing your release
- When done, run:

    git flow release finish '1.0.0'

PS C:\works> echo "1.0.0" >> version
PS C:\works> git add .
PS C:\works> git commit -m "chore(main): add version"
[release/1.0.0 ba6dc8b] chore(main): add version
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 version
PS C:\works> git flow release finish -m "ver 1" 1.0.0
```

Figure 6: Инициализация начала и завершения релиза ветки

Использование системы управления версиями

```
PS C:\work> git push --all
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 474 bytes | 237.00 KiB/s, done.
Total 5 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 1 local object.
To github.com:timbow64/MatMod.git
 * [new branch]      master -> master
 * [new branch]      develop -> develop
PS C:\work> git push --tags
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 158 bytes | 52.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:timbow64/MatMod.git
 * [new tag]         v1.0.0 -> v1.0.0
PS C:\work>
```

Figure 7: Завершаем релиз. Отправляем список изменений в сетевой репозиторий

Выполним объединение веток

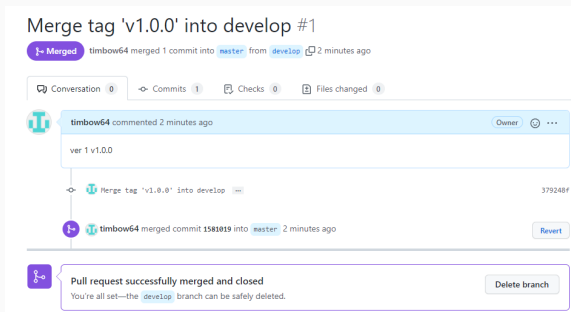


Figure 8: Объединение веток в сетевом репозитории

Выводы по проделанной работе

Мы изучили и применили средства контроля версий git, а также научились работать с различными командами в консоли.