Создание репозитория

Из локального репозитория в удаленный

curl -u 'USER' https://api.github.com/user/repos -d '{"name":"REPO"}'

git remote add origin [git@github.com:USER/REPO.git](mailto:git@github.com:USER/REPO.git)

git push origin master

{

"name": "<Repo Name>",

"description": "<Some message>",

"homepage": "https://github.com",

"private": false,

}

**git init**

**curl -u 'user\_name:password' https://api.github.com/user/repos -d '{"name":"wp1.loc"}'**

**git remote add origin** [**git@github.com:user\_name/wp1.loc.git**](mailto:git@github.com:Alex-Smil/wp1.loc.git)

Далее git add . – git commit и т.д.

**git push -u origin master**

Если git push не сработал и вывелась ошибка:

git@github.com: Permission denied (publickey).

fatal: Could not read from remote repository.

Это значит, что надо настроить ключи:

Generating a new SSH key:

1. Open Git Bash.
2. Paste the text below, substituting in your GitHub email address.

$ **ssh-keygen -t rsa -b 4096 -C "your\_email@example.com"**

This creates a new ssh key, using the provided email as a label.

> Generating public/private rsa key pair.

1. When you're prompted to "Enter a file in which to save the key," press Enter. This accepts the default file location.

> Enter a file in which to save the key (/c/Users/you/.ssh/id\_rsa):[Press enter]

Если не указывать путь куда будет сохранен ключ, то по умолч.сохраняет в C:\Users\Smile\.ssh

1. At the prompt, type a secure passphrase. For more information, see ["Working with SSH key passphrases"](https://help.github.com/en/articles/working-with-ssh-key-passphrases).
2. > Enter passphrase (empty for no passphrase): [Type a passphrase]

> Enter same passphrase again: [Type passphrase again]

Ensure the ssh-agent is running:

**eval `ssh-agent -s`**

**ssh-add ~/.ssh/id\_rsa**

Add a new SSH key to your GitHub account:

В разделе Settings - SSH and GPG keys - New SSH key – в поле key вставляем ключ из файла id\_rsa.pub (по умолч. См в папке C:\Users\Саша\.ssh) - Add SSH key

В папке C:\Users\Саша\.ssh

Мое ключ. слово

> Enter passphrase (empty for no passphrase): [Type a passphrase] - key\_word

> Enter same passphrase again: [Type passphrase again] - key\_word

Generating a new SSH key and adding it to the ssh-agent - <https://help.github.com/en/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent>

<https://stackoverflow.com/questions/17846529/could-not-open-a-connection-to-your-authentication-agent>

<https://stackoverflow.com/questions/30816966/github-permission-denied-publickey-but-i-got-youve-successfully-authenticat>

**Создание репозитория**

**Клонирование удаленного (уже существующего) репозитория на локальный**

1. создаю удаленный репозиторий

2. клонирую удаленный репозиторий в локальную папку.

git clone https://github.com/user\_name/GitHub-experiments-Repo.git

3. наполняем папку далее

git add name\_file

git git commit -m"text of message"

\*\*\*\*\* git add

git add "\*.txt"

\*\*\*\*\* посмотреть свои коммиты, логи изменений

git log

выход из лог журнала q

выборка из лога, например:

болле подробно:

git log --pretty=format:"%h - %an, %ar : %s"

коммиты за отрезок времени

git log --since=2.weeks

из helpa file:///C:/Program%20Files/Git/mingw64/share/doc/git-doc/git-log.html

git log --since="120 minutes ago"

git log --since="5 minutes ago"

с указанием кол-ва подробных коммитов

-2 в данном случае кол-во коммитов, можно сколко угодно

git log -p -2

просмотр файлов в текущей директории в консоли

dir

Без git

\*\*\*\*\*\*\*\*\*\*\* push \*\*\*\*\*\*\*\*\*\*\*\*

git push -u origin master (если первый раз , настройку глянуть у афансьева)

git push (все последющие разы)

\*\*\*\*\* Игнорирование файлов в папке

создаем файл .gitignore - имя можно любое, но с точкой в начале

с расширением All types (\*.\*)

внутри пишем:

# комментарий

# проигнорировать папку

# dir/

# проигнорировать файлы с определенным расширением в паке dir

# dir/\*.txt

ignoreFolder/

testProject\_Ignore1.html

\*\*\* например\*/

\*\*\*\*\* Откат внесенных изменений к последнему коммиту, например я писал-писал и понял что все не то, и хочу вернуться назад, тогда

$ git checkout -- file.txt

\*\*\*\*\* Отмена/перезапись предыдущего комиита

git commit --amend -m"новый комментарий"

https://git-scm.com/book/ru/v2/%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D1%8B-Git-%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%B8-%D0%BE%D1%82%D0%BC%D0%B5%D0%BD%D1%8B

\*\*\*\*

Назначение своего редактора для коммитов

git config --global core.editor "'C:\Program Files (x86)\Notepad++\notepad++.exe' -multiInst -notabbar -nosession -noPlugin"

Создание веток

Создаем ветку и сразу переходим в нее  
git checkout -b name\_newBranch

Проверяем существующие ветки  
git branch

git branch -a

git branch -v  
так будут видны последние коммиты

создание ветки без перехода в нее

git branch name\_of\_newBranch

переход между ветками  
git checkout name\_of\_newBranch

Удаление ветки

**локальной:**  
git branch -d name\_branch

**удаленка:**

git push origin --delete name\_branch

Слияние веток

Перед тем как залить ветку newBranch в ветку master во избежание ситуации, что пока мы делали ветку newBranch в ветке master появились новые коммиты несовместимые изменения с нашей новой feature, сначала мы заливаем свежий master в ветку newBranch, проверяем и только потом заливаем ветку newBranch в ветку master.

https://youtu.be/GVpPV4tHT58?list=PLoonZ8wII66iUm84o7nadL-oqINzBLk5g

**merge**

https://youtu.be/xAKnRuYobdc?list=PLoonZ8wII66iUm84o7nadL-oqINzBLk5g&t=127

merge делаем из ветки в которую будем заливать другую

git merge name\_branch

git merge master

========================

https://youtu.be/xAKnRuYobdc?list=PLoonZ8wII66iUm84o7nadL-oqINzBLk5g&t=189

git config --global merge.tool kdiff3

клевая утилита

если kdiff не установлен, то качаем его из нета  
потом вводим команду

git config --global merge.tool.kdiff3.cmd '" C:\\Program Files\\KDiff3\\kdiff3" $BASE $GLOBAL $REMOTE -o $MERGED'

C:\Program Files\KDiff3

==========================