**Шпаргалка. Базовые команды в консоли**

Чтобы вам было удобнее взаимодействовать с командной строкой, мы подготовили шпаргалку. В ней собраны все команды, о которых мы рассказали в уроках, и их полезные вариации.

**Навигация**

* pwd (от англ. ***p****rint* ***w****orking* ***d****irectory*, «показать рабочую папку») — покажи, в какой я папке;
* ls (от англ. ***l****i****s****t directory contents*, «отобразить содержимое директории») — покажи файлы и папки в текущей папке;
* ls -a — покажи также скрытые файлы и папки, названия которых начинаются с символа .;
* cd first-project (от англ. ***c****hange* ***d****irectory*, «сменить директорию») — перейди в папку first-project;
* cd first-project/html — перейди в папку html, которая находится в папке first-project;
* cd .. — перейди на уровень выше, в родительскую папку;
* cd ~ — перейди в домашнюю директорию (/Users/Username);
* cd / — перейди в корневую директорию.

**Работа с файлами и папками**

**Создание**

* touch index.html (англ. *touch,* «коснуться») — создай файл index.html в текущей папке;
* touch index.html style.css script.js — если нужно создать сразу несколько файлов, можно напечатать их имена в одну строку через пробел;
* mkdir second-project (от англ. ***m****a****k****e* ***dir****ectory*, «создать директорию») — создай папку с именем second-project в текущей папке.

**Копирование и перемещение**

* cp file.txt ~/my-dir (от англ. ***c****o****p****y*, «копировать») — скопируй файл в другое место;
* mv file.txt ~/my-dir (от англ. ***m****o****v****e*, «переместить») — перемести файл или папку в другое место.

**Чтение**

* cat file.txt (от англ. *con****cat****enate and print*, «объединить и распечатать») — распечатай содержимое текстового файла file.txt.

**Удаление**

* rm about.html (от англ. ***r****e****m****ove*, «удалить») — удали файл about.html;
* rmdir images (от англ. ***r****e****m****ove* ***dir****ectory*, «удалить директорию») — удали папку images;
* rm -r second-project (от англ. ***r****e****m****ove,* «удалить» + ***r****ecursive*, «рекурсивный») — удали папку second-project и всё, что она содержит.

**Полезные возможности**

* Команды необязательно печатать и выполнять по очереди. Можно указать их списком — разделить двумя амперсандами (&&).
* У консоли есть собственная память — буфер с несколькими последними командами. По ним можно перемещаться с помощью клавиш со стрелками вверх (**↑**) и вниз (**↓**).
* Чтобы не вводить название файла или папки полностью, можно набрать первые символы имени и дважды нажать Tab. Если файл или папка есть в текущей директории, командная строка допишет путь сама.

Например, вы находитесь в папке dev. Начните вводить cd first и дважды нажмите Tab. Если папка first-project есть внутри dev, командная строка автоматически подставит её имя. Останется только нажать Enter.

# Что такое SSH. Генерируем SSH-ключ

Один из наиболее распространённых сетевых протоколов — **SSH** (от англ. ***S***ecure ***Sh***ell Protocol).

SSH использует пару ключей для обеспечения безопасности — публичный и приватный:

* **Приватный ключ** (англ. *private key*) хранится только на вашем компьютере и не должен передаваться кому-либо ещё. Он используется для расшифровки данных.
* **Публичный ключ** (англ. *public key*) доступен всем и используется для шифрования данных. Они могут быть расшифрованы парным приватным ключом.

### Проверка наличия SSH-ключа

$ ls -la .ssh/ # вывели список созданных ключей

Если есть файлы с похожими названиями, SSH-ключи уже создавались:

* id\_dsa.pub;
* id\_ecdsa.pub;
* id\_ed25519.pub;
* id\_rsa.pub.

**Инструкция по генерации SSH-ключа**

1. Для генерации SSH-пары можно использовать программу ssh-keygen. Откройте терминал и введите следующую команду.

Скопировать кодBASH

$ ssh-keygen -t ed25519 -C "электронная почта, к которой привязан ваш аккаунт на GitHub"

Используйте электронную почту, к которой привязан ваш GitHub-аккаунт.

Если вы видите сообщение об ошибке, то, скорее всего, ваша система не поддерживает алгоритм шифрования ed25519. Ничего страшного: используйте другой алгоритм.

Скопировать кодBASH

$ ssh-keygen -t rsa -b 4096 -C "электронная почта, к которой привязан ваш аккаунт на GitHub"

После ввода отобразится такое сообщение.

Скопировать кодBASH

> Generating public/private rsa key pair. *# сгенерированы публичный и приватный ключи*

1. Укажите место хранения ключей. Простой вариант — сделать домашний каталог пользователя путём по умолчанию. Для этого нажмите Enter.

**macOS**

Скопировать кодBASH

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

**Windows**

Скопировать кодBASH

> Enter a file in which to save the key (C:\Users\<имя\_пользователя>\.ssh\):[Press enter]

Теперь в указанной директории появится пара ключей.

1. Программа запросит **кодовую фразу** (англ. *passphrase*) для доступа к SSH-ключу. Вы можете оставить поле пустым. Для этого нажмите Enter, а затем ещё раз Enter для подтверждения.

Скопировать кодBASH

> Enter passphrase (empty for no passphrase): [Type a passphrase]

> Enter same passphrase again: [Type passphrase again]

💡 **Быть или не быть кодовой фразе — вот в чём вопрос**

Как бы странно ни звучало, кодовая фраза — это «пароль от ключа». Представьте, что SSH-ключ лежит в шкатулке. А на самой шкатулке — кодовый замок, который открывается кодовой фразой.

Многие пользователи Git не используют кодовую фразу для защиты своего SSH-ключа. Если такой фразы нет, то её не нужно вводить всякий раз при взаимодействии с удалённым репозиторием.

С другой стороны, применение кодовой фразы усиливает безопасность ключей. Если вы используете эту фразу, ключ будет надёжно защищён в случае несанкционированного доступа к вашему компьютеру.

1. Готово! Теперь осталось проверить, что ключи действительно сгенерировались. Для этого вызовите эту команду.

Скопировать кодBASH

ls -a ~/.ssh

На экране должны появиться два файла — один с расширением .pub, другой — без. Файл в .pub — публичный, им можно делиться с веб-сайтами или коллегами. Файл без расширения .pub — приватный. Ни в коем случае не передавайте его никому!

Вся последовательность действий в консоли показана на скриншоте ниже.



**Инструкция по связыванию SSH-ключа и GitHub-аккаунта**

1. После выполнения команды ssh-keygen из предыдущего урока в директории ~/.ssh будет создано два файла — id\_ed25519 и id\_ed25519.pub (или id\_rsa и id\_rsa.pub — в зависимости от того, какой алгоритм вы использовали):
   * id\_ed25519/id\_rsa — приватный ключ (файл без .pub в конце). Ни в коем случае не копируйте его и не делитесь им.
   * id\_ed25519.pub/id\_rsa.pub — публичный ключ (на это указывает расширение .pub).

Скопируйте содержимое файла с публичным ключом в буфер обмена.

**Windows**

Скопировать кодBASH

*# скопировать содержимое ключа в буфер обмена:*

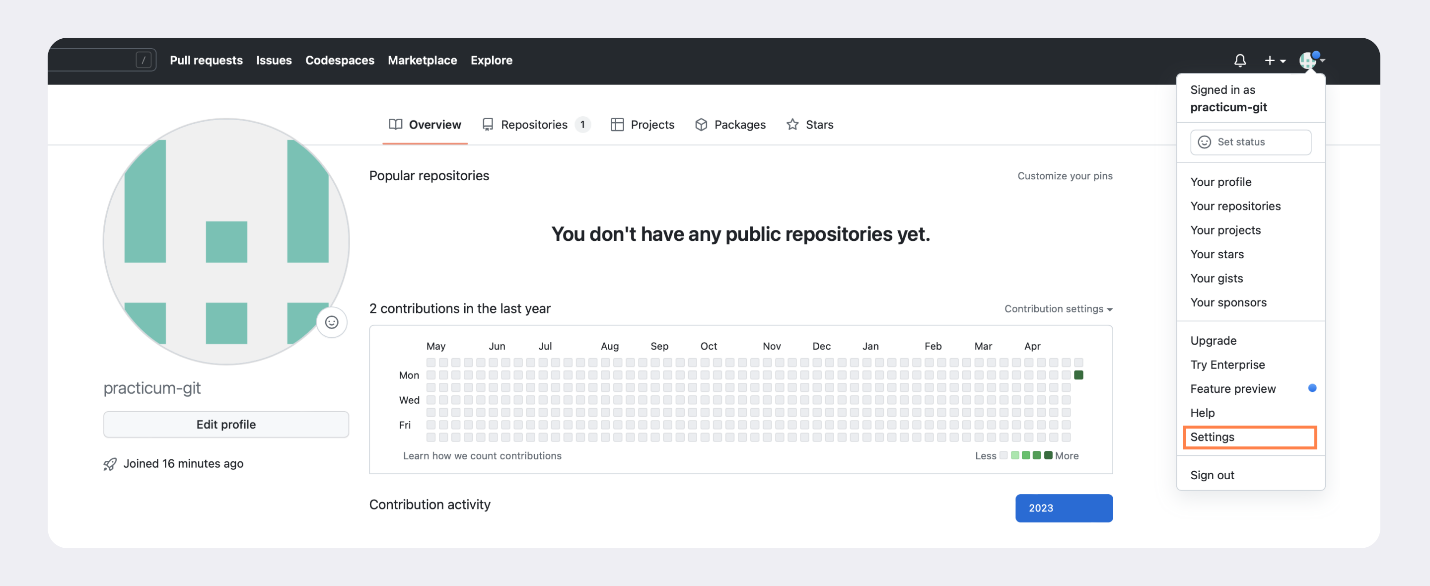
$ clip < ~/.ssh/id\_rsa.pub

*# для ed25519:*

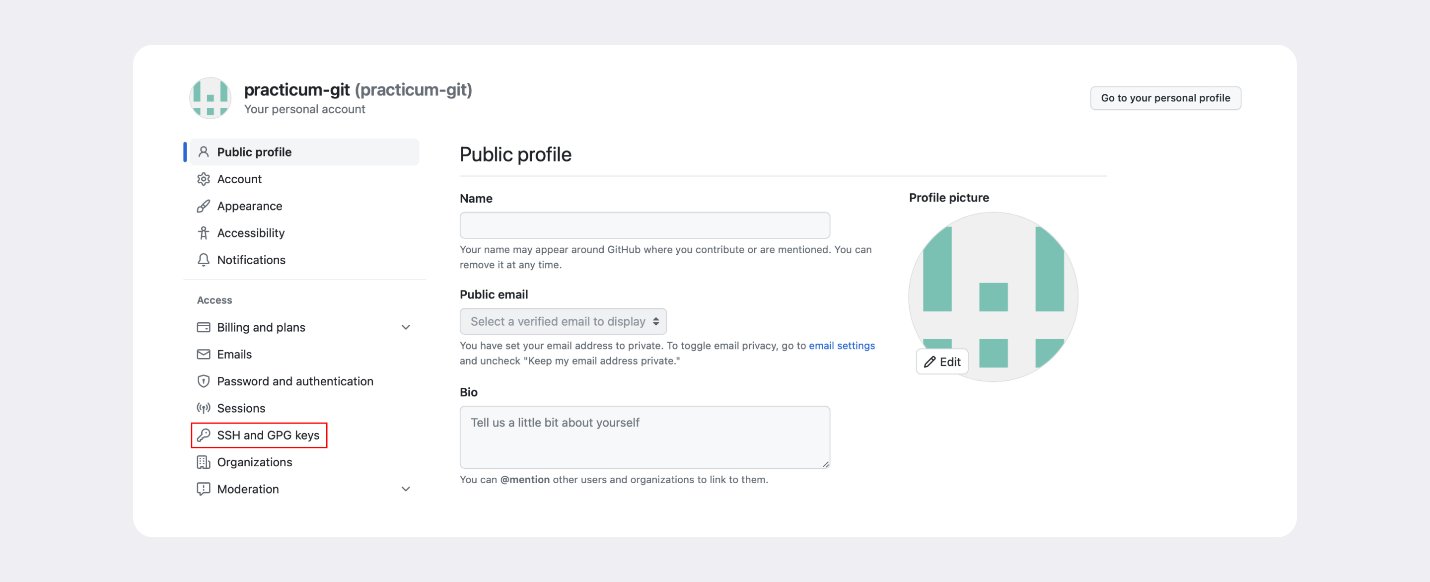
$ clip < ~/.ssh/id\_ed25519.pub

Если clip не сработает, выведите содержимое файла с помощью cat ~/.ssh/id\_rsa.pub или cat ~/.ssh/id\_ed25519.pub и скопируйте вывод в буфер обмена из консоли.

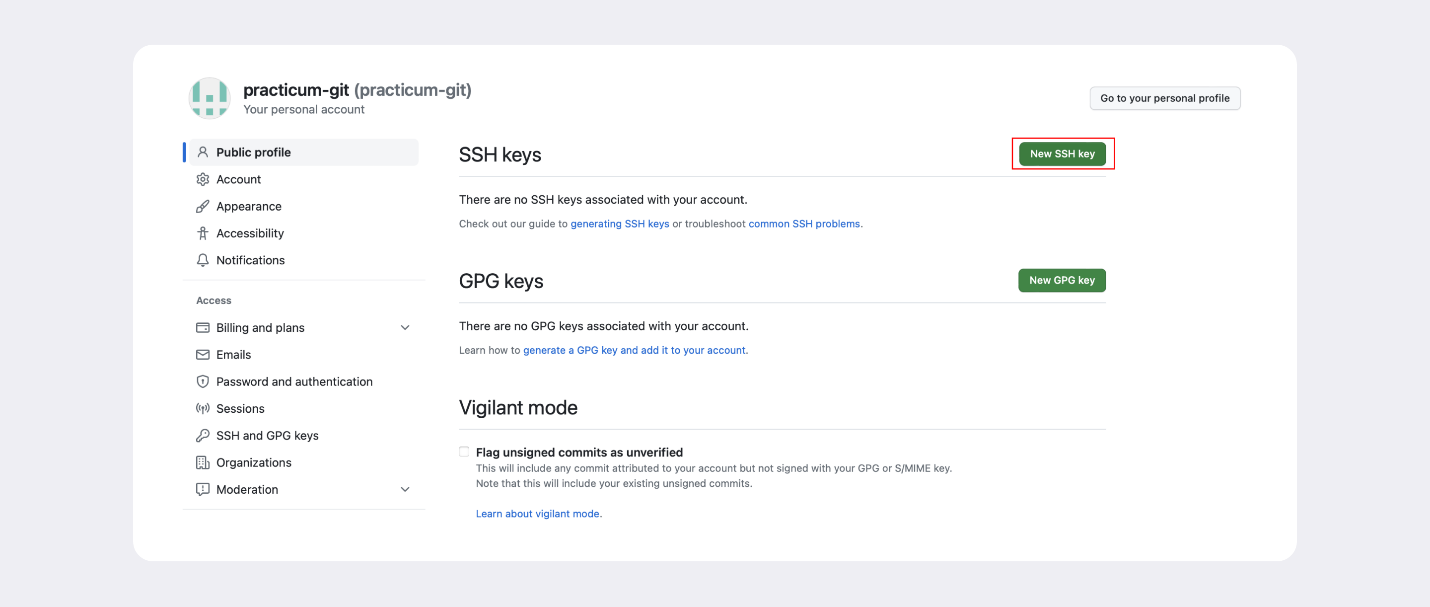
1. Перейдите на GitHub и выберите пункт **Settings** (англ. «настройки») в меню аккаунта.



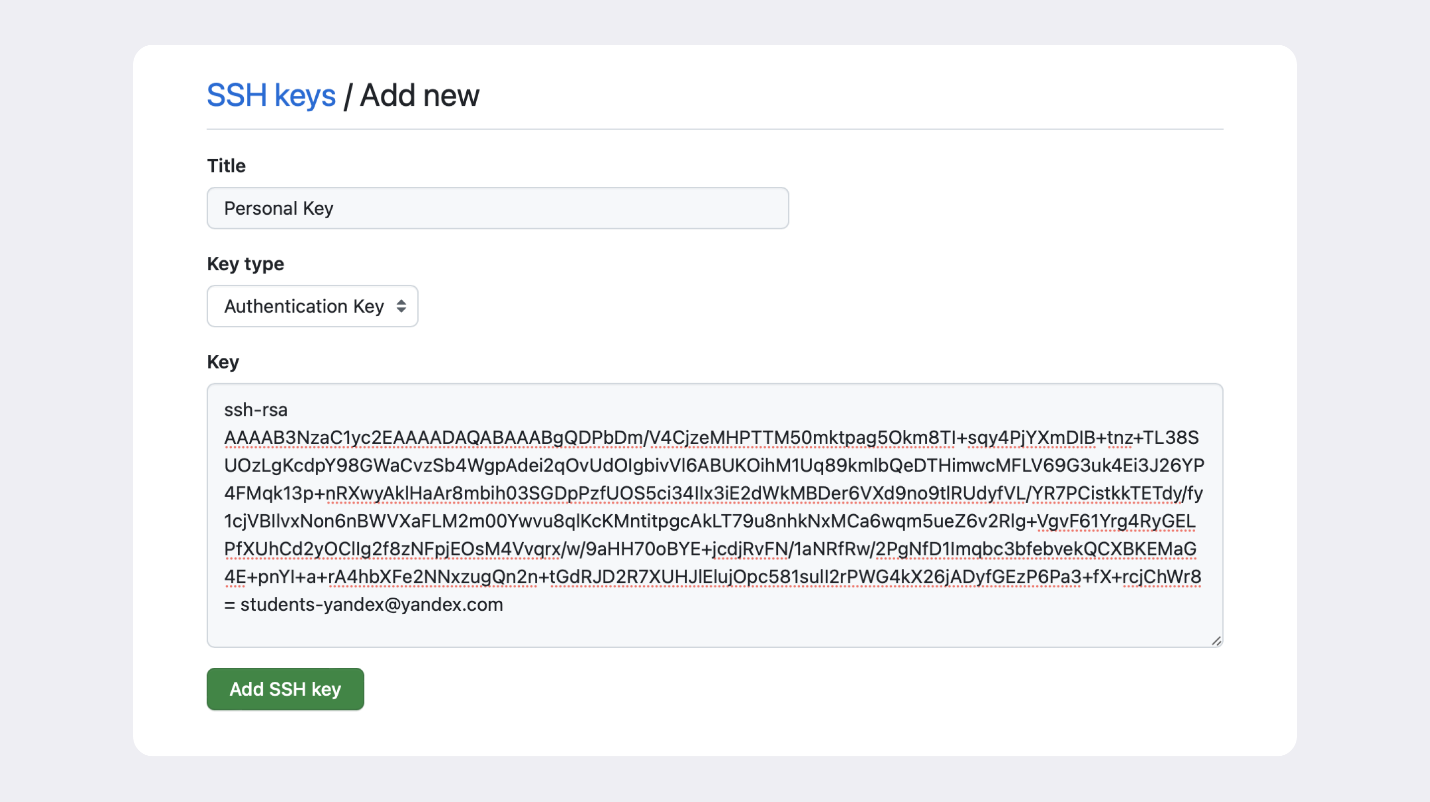
1. В меню слева нажмите на пункт **SSH and GPG keys**.



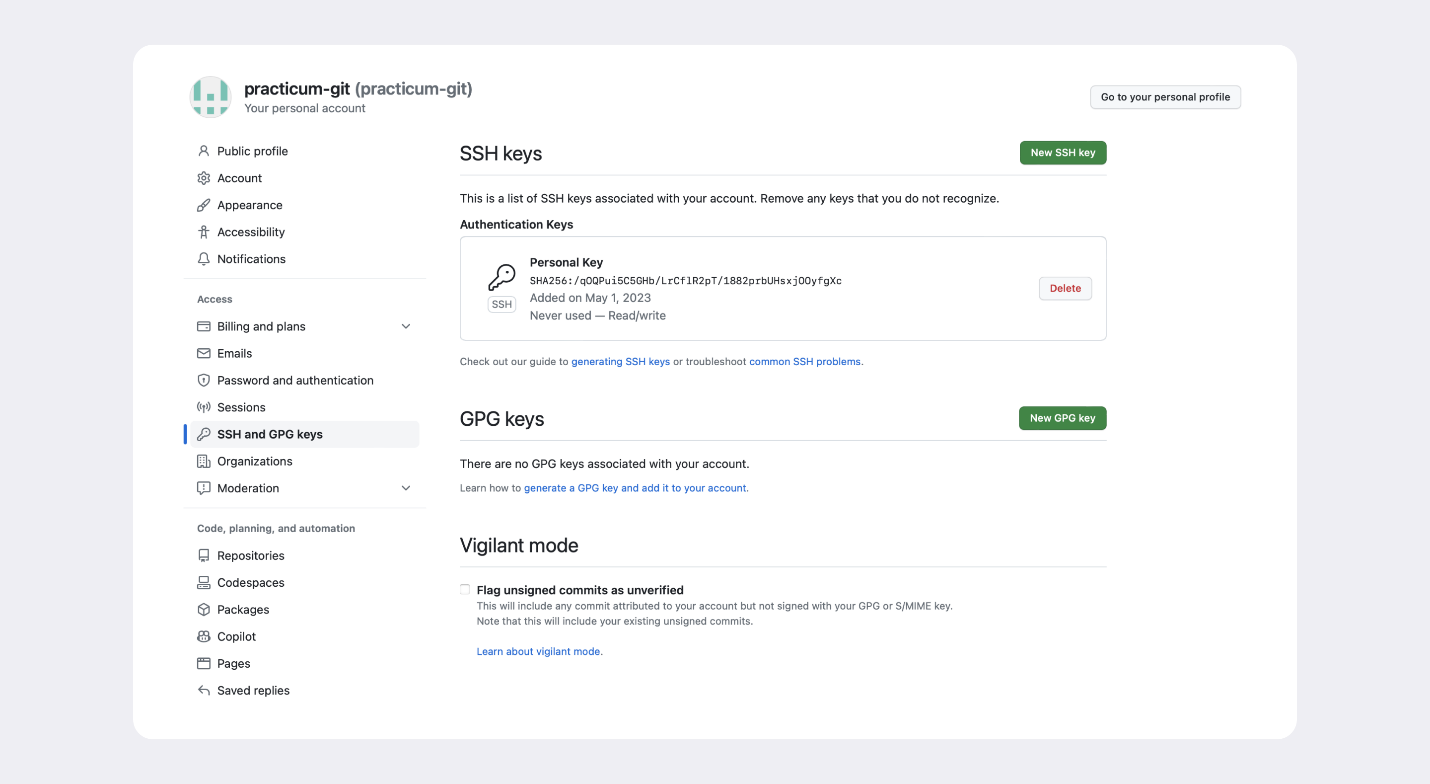
1. В открывшейся вкладке выберите **New SSH key** (англ. «новый SSH-ключ»).



1. В поле **Title** (англ. «заголовок») напишите название ключа. Например, **Personal key** (англ. «личный ключ»).
2. В поле **Key type** (англ. «тип ключа») должно быть **Authentication Key** (англ. «ключ аутентификации»).
3. В поле **Key** скопируйте ваш ключ из буфера обмена.



8. Нажмите на кнопку **Add SSH key** (англ. «добавить SSH-ключ»).



1. Проверьте правильность ключа с помощью следующей команды.

$ ssh -T git@github.com

Если это первый раз, когда вы используете Git, чтобы поделиться проектом на GitHub, появится похожее предупреждение.

The authenticity of host 'github.com (140.82.121.4)' can't be established. ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU. This key is not known by any other names. Are you sure you want to continue connecting (yes/no/[fingerprint])?

Это предупреждение сообщает, что вы никогда не соединялись с сервером GitHub. Поэтому Git не может гарантировать, что сервер является тем, за кого он себя выдаёт.

Для подтверждения подлинности сервер генерирует и публикует ключи SHA256. Вы можете проверить ключи GitHub [по этой ссылке](https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/githubs-ssh-key-fingerprints). Если ключ в предупреждении совпадает с тем, что вы видите на сайте, значит, сервер является действительным. Введите yes, чтобы продолжить. Вы увидите приветствие на экране.

Hi %ВАШ\_АККАУНТ%! You've successfully authenticated, but GitHub does not provide shell access

Команды GIT

Админ ПЭК@PC222-00 MINGW64 ~/Documents/Git Command

$ git init

Initialized empty Git repository in C:/Users/Админ ПЭК/Documents/Git Command/.git/

Админ ПЭК@PC222-00 MINGW64 ~/Documents/Git Command (master)

$ git add .

Админ ПЭК@PC222-00 MINGW64 ~/Documents/Git Command (master)

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: "\320\232\320\276\320\274\320\260\320\275\320\264\321\213 Git.docx"

Админ ПЭК@PC222-00 MINGW64 ~/Documents/Git Command (master)

$ git commit -m "file on github"

Author identity unknown

\*\*\* Please tell me who you are.

Run

git config --global user.email "you@example.com"

git config --global user.name "Your Name"

to set your account's default identity.

Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'Админ ПЭК@PC222-00.(none)')

Админ ПЭК@PC222-00 MINGW64 ~/Documents/Git Command (master)

$ git config --global user.email "kee\_lidamt@mail.ru"

Админ ПЭК@PC222-00 MINGW64 ~/Documents/Git Command (master)

$ git commit -m "file on github" [master (root-commit) 2eedfec] file on github

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 "\320\232\320\276\320\274\320\260\320\275\320\264\321\213 Git.docx"

Админ ПЭК@PC222-00 MINGW64 ~/Documents/Git Command (master)

$ git remote add origin git@github.com:ElenaKuzmina/Example270124.git

Админ ПЭК@PC222-00 MINGW64 ~/Documents/Git Command (master)

$ git status

On branch master

nothing to commit, working tree clean

Админ ПЭК@PC222-00 MINGW64 ~/Documents/Git Command (master)

$ git push -u origin master

Enumerating objects: 3, done.

Counting objects: 100% (3/3), done.

Delta compression using up to 4 threads

Compressing objects: 100% (3/3), done.

Writing objects: 100% (3/3), 1.12 MiB | 6.04 MiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

To github.com:ElenaKuzmina/Example270124.git

\* [new branch] master -> master

branch 'master' set up to track 'origin/master'.

Админ ПЭК@PC222-00 MINGW64 ~/Documents/Git Command (master)

$