

# Справочник по Git

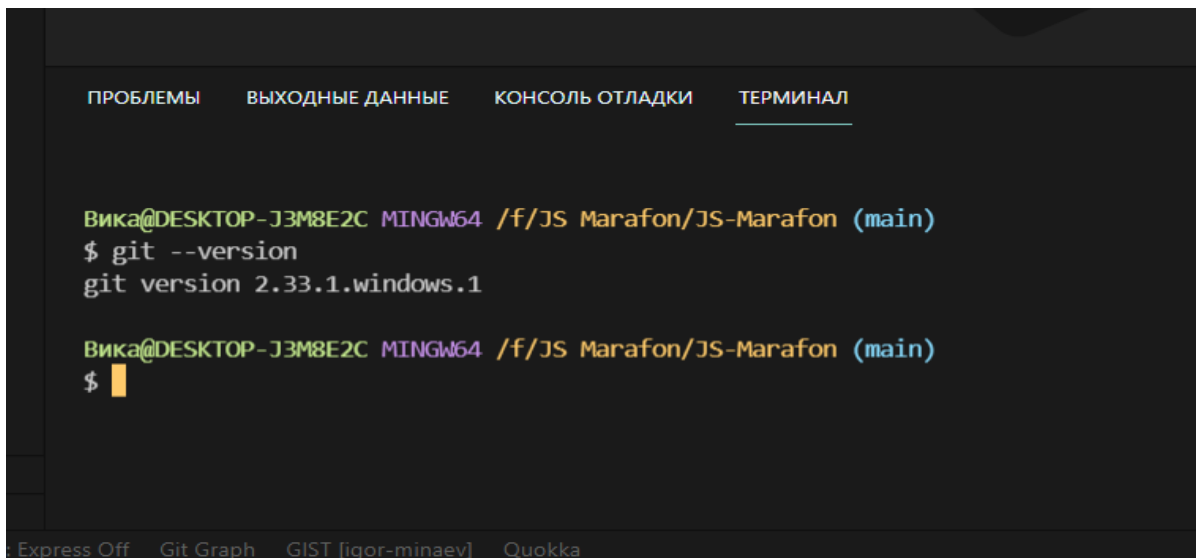
**Git** — распределённая система контроля версий, которая дает возможность разработчикам отслеживать изменения в файлах и работать над одним проектом совместно с коллегами.

## Установка Git для Windows

Устанавливаем Git на свой компьютер - [git for windows](#)

Для проверки правильности установки и версии Git необходимо использовать команду **git --version**

Терминал VS Code: Терминал -> Создать терминал -> Вводим **git --version**

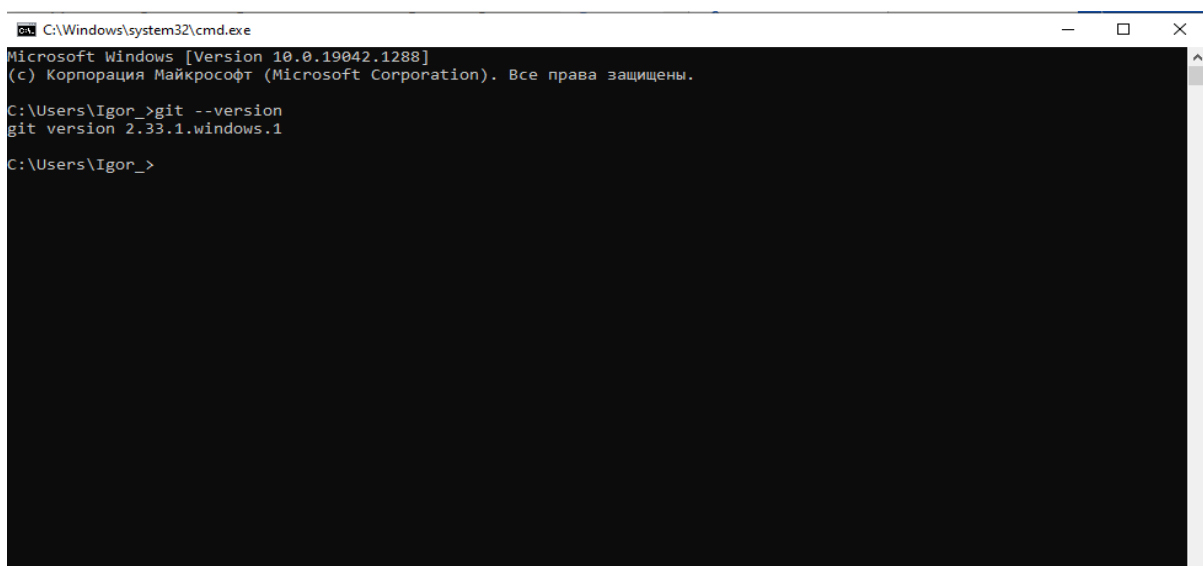


```
ПРОБЛЕМЫ    ВЫХОДНЫЕ ДАННЫЕ    КОНСОЛЬ ОТЛАДКИ    ТЕРМИНАЛ

Вика@DESKTOP-J3M8E2C MINGW64 /f/JS Marafon/JS-Marafon (main)
$ git --version
git version 2.33.1.windows.1

Вика@DESKTOP-J3M8E2C MINGW64 /f/JS Marafon/JS-Marafon (main)
$
```

Терминал Windows: Сочетание клавиш Win + R -> В появившемся окне вводим cmd -> В терминале Windows вводим **git --version**



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19042.1288]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\Igor_>git --version
git version 2.33.1.windows.1

C:\Users\Igor_>
```

## Установка имени и электронной почты

Если вы никогда ранее не использовали git, для начала вам необходимо осуществить установку. Выполните следующие команды, чтобы git узнал ваше имя и электронную почту.

```
git config --global user.name "Your Name" - имя пользователя
```

```
git config --global user.email your\_email@whatever.com - email пользователя
```

Чтобы посмотреть текущие данные пользователя:

```
git config --global user.name
```

```
git config --global user.email
```

## Аккаунт на Github

Также для работы понадобится создать аккаунт на [Github](https://github.com) — это бесплатный (для одиночного использования) сервис, в котором хранят свои проекты большинство компаний и разработчиков.

## Создание репозитория

Для создания локального репозитория используется команда: `git init`

## Отмена изменений до добавления

Для отмены изменений до добавления в локальный репозиторий используется команда:

```
git checkout -- имя файла - определенный файл
```

```
git checkout . - все файлы
```

## Добавление изменений

Для добавления изменений в локальный репозиторий используется команда:

```
git add "имя файла" - определенный файл
```

```
git add . - все файлы
```

## Отмена изменений после добавления

Для отмены изменений после добавления в локальный репозиторий используется команда:

`git reset имя файла` - определенный файл

`git reset .` - все файлы

## Просмотр статуса репозитория

Для просмотра статуса локального репозитория используется команда:

`git status`

## Создание коммита

Для добавления коммита в локальный репозиторий используется команда:

`git commit -m "название коммита"`

## Отмена последнего коммита и удаление всех изменений этого коммита

Для отмены последнего коммита с удалением всех его изменений в локальном репозитории используется команда:

`git reset --hard HEAD^1`

## Отмена последнего коммита и без удаление всех изменений этого коммита

Для отмены последнего коммита без удаления всех его изменений в локальном репозитории используется команда:

`git reset --soft HEAD^1`

## Просмотр истории

Для просмотра истории коммитов в локальном репозитории используется команда:

`git log`

## Работа с ветками

Для просмотра всех веток в локальном репозитории используется команда:

`git branch` - показывает все ветки (\* указывается ветка, в которой мы находимся)

`git branch -v` - показывает ветку и последний коммит в этой ветке

Для создания ветки используется команда: `git branch название ветки`

Для перехода из одной ветки в другую используется команда:

`git checkout название ветки`

Для создания ветки и перехода в нее можно использовать команду:

`git checkout -b название ветки`

Для переименования ветки используется команда:

`git branch -m новое название`

Для удаления ветки используется команда:

`git branch -D название ветки`

Для слияния веток:

1. Переходим на ветку, в которую хотим залить изменения, например это ветка develop

2. В ветке develop вводим команду `git merge имя ветки, с которой стягиваем код` (например: `git merge master`). В итоге весь код из ветки master будет теперь и ветке develop.

## Отправка проекта на удаленный репозиторий

Для начала необходимо создать репозиторий на Github, затем для того, чтобы залить наш проект на удаленный репозиторий используется команда:

`git remote add origin ссылка на наш репозиторий` - привязываем наш локальный репозиторий к удаленному(проделывается 1 раз)

`git push -u origin master` - отправляем все наши изменения на удаленный репозиторий

В дальнейшем для отправки изменений используем команду:

`git push`

Для того чтобы клонировать удаленный репозиторий себе на компьютер используем команду: `git clone`

При работе командой в одном репозитории другие разработчики так же будут отправлять свои изменения на удаленный репозиторий, чтобы их клонировать себе на компьютер используется команда: `git pull`

