

Быстрый старт с Git: установка, настройка и терминал

Вот краткая, пошаговая инструкция для новичков — от установки Git до первых шагов в терминале.

1) Установка Git на разных ОС

Windows (2 пути)

А. Git Bash (рекомендовано для новичков)

1. Скачайте установщик с сайта git-scm.com → запустите.
2. На шагах мастера можно оставить почти всё по умолчанию. Обратите внимание:
 - **Editor**: выберите ваш любимый (например, “Visual Studio Code”).
 - **Adjust PATH**: “Use Git from the command line...”.
 - **Line endings**: “Checkout Windows-style, commit Unix-style (recommended)”.
3. После установки откройте **Git Bash** (ищите в меню Пуск).

Б. Установка через пакетный менеджер

- **winget** (Windows 10/11): откройте PowerShell и выполните:

```
winget install --id Git.Git -e
```

macOS

Вариант 1 (быстрее всего)

Откройте Терминал и выполните:

```
xcode-select --install
```

(установит инструменты разработчика, включая git)

Вариант 2 (через Homebrew)

```
/bin/bash -c "$(curl -fsSL  
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

```
brew install git
```

Linux

Откройте терминал и выполните команду для вашего дистрибутива:

- Debian/Ubuntu:

```
sudo apt update && sudo apt install -y git
```

- Fedora:

```
sudo dnf install -y git
```

- Arch:

```
sudo pacman -S git
```

- openSUSE:

```
sudo zypper install -y git
```

Проверка установки (везде):

```
git --version
```

2) Базовый конфиг Git

Выполните **один раз** (подставьте свои данные):

```
git config --global user.name "Ваше Имя"
git config --global user.email "ваш.email@example.com"

# Рекомендуемые настройки
git config --global init.defaultBranch main
git config --global pull.rebase false
git config --global color.ui auto

# Редактор по умолчанию (варианты)
git config --global core.editor "code --wait" # VS Code
```

```
# или
git config --global core.editor "nano"

# Переводы строк (важно для Windows)
# Windows:
git config --global core.autocrlf true
# macOS/Linux:
git config --global core.autocrlf input
```

Посмотреть весь текущий конфиг:

```
git config --global -l
```

3) Генерация SSH-ключей и добавление в GitHub

3.1 Создаём ключ

Рекомендуется **ed25519**:

```
ssh-keygen -t ed25519 -C "ваш.email@example.com"
```

- Когда спросят путь — нажмите **Enter** (по умолчанию `~/.ssh/id_ed25519`).
- Введите **пароль-фразу** (желательно, можно пустую — но лучше задать).

3.2 Запускаем ssh-agent и добавляем ключ

macOS / Linux / Git Bash (Windows):

```
eval "$(ssh-agent -s)"
ssh-add ~/.ssh/id_ed25519
```

Windows PowerShell (встроенный OpenSSH):

```
Get-Service ssh-agent | Set-Service -StartupType Automatic
Start-Service ssh-agent
ssh-add $env:USERPROFILE\.ssh\id_ed25519
```

3.3 Копируем публичный ключ в буфер обмена

- macOS:

```
pbcopy < ~/.ssh/id_ed25519.pub
```

- **Linux (если нет xclip — установите):**

```
xclip -selection clipboard < ~/.ssh/id_ed25519.pub
```

- **Windows PowerShell:**

```
Get-Content $env:USERPROFILE\.ssh\id_ed25519.pub | Set-Clipboard
```

- **Git Bash (Windows):**

```
clip < ~/.ssh/id_ed25519.pub
```

3.4 Добавляем ключ в GitHub

1. Зайдите в GitHub → **Settings** → **SSH and GPG keys** → **New SSH key**.
2. Вставьте ключ (поле **Key**), задайте имя (например, “My Laptop”), сохраните.

3.5 Проверяем связь по SSH

```
ssh -T git@github.com
```

Ожидаемо увидите приветствие вроде: *“Hi <ваш_логин>! You’ve successfully authenticated...”*.

4) Как открыть терминал и установить Git Bash

- **Linux:** чаще всего **Ctrl + Alt + T**; или найдите “Terminal”/“Konsole”/“GNOME Terminal” в меню приложений.
- **macOS:**
 - Spotlight: **Cmd + Space** → введите “Terminal” → Enter.
 - Или: **Applications** → **Utilities** → **Terminal**.
 - (Необязательно: установите **iTerm2** — удобный терминал.)
- **Windows (Git Bash):**
 - Установите с **git-scm.com** (см. раздел 1).

- Откройте **Git Bash** через меню Пуск или ПКМ в папке → “Git Bash Here”.
 - **Windows (PowerShell/Terminal):**
 - **Win + X** → **Windows Terminal** или “PowerShell”.
 - В Windows Terminal можно добавить профиль “Git Bash”.
-

5) Топ-10 команд терминала (с примерами)

Ниже — команды в стиле **bash** (Linux/macOS/Git Bash).

В PowerShell многие алиасы работают (например, `ls`, `cat`), но родные команды отличаются.

1. `pwd` — показать текущую папку:

```
pwd
```

2. `ls` — список файлов:

```
ls -la      # подробный список, включая скрытые
```

3. `cd` — переход между папками:

```
cd projects/myapp
cd ..      # на уровень вверх
```

4. `mkdir` — создать папку:

```
mkdir src
mkdir -p a/b/c  # создать вложенные
```

5. `touch` — создать/обновить файл (в PowerShell аналог: `New-Item`):

```
touch README.md
```

6. `cp` — копировать:

```
cp file.txt backup.txt
cp -r src src_backup  # рекурсивно папку
```

7. `mv` — переместить/переименовать:

```
mv oldname.txt newname.txt
mv file.txt ../          # перенести уровнем выше
```

8. `rm` — удалить:

```
rm file.txt
rm -r build/          # удалить папку рекурсивно
rm -rf build/         # осторожно: без подтверждения!
```

9. `cat` / `less` — посмотреть содержимое:

```
cat file.txt
less long.log        # прокрутка, q — выход
```

10. `grep` — поиск по тексту:

```
grep "TODO" -nR .
# -n: номера строк, -R: рекурсивно по папкам
```
