怎么使用git来管理自己的代码。我想把我的代码存到github

太好了！使用 **Git + GitHub** 是管理个人代码、备份项目、展示作品的最佳方式。下面我会用最清晰、实用的方式，带你从零开始学会如何用 Git 把你的代码上传到 GitHub。

**🌟 一、准备工作**

**✅ 1. 安装 Git**

如果你还没安装，请先下载并安装 Git：

👉 https://git-scm.com/download/win

安装时一路“下一步”，默认选项即可。

安装完成后打开命令行（CMD 或 PowerShell），输入：

bash

编辑

git --version

如果显示版本号（如 git version 2.40.1），说明安装成功。

**✅ 2. 注册 GitHub 账号**

前往官网注册一个免费账号：

👉 https://github.com/

推荐使用邮箱注册，并完成邮箱验证。

**🛠️ 二、配置 Git（只做一次）**

在命令行中运行以下命令，设置你的用户名和邮箱（这是你提交代码的“身份”）：

bash

编辑

git config --global user.name "你的名字"

git config --global user.email "你的邮箱"

📌 示例：

bash

编辑

git config --global user.name "ZhangSan"

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

✅ 邮箱建议与 GitHub 注册邮箱一致。

**📁 三、创建本地项目并初始化 Git**

假设你想上传的代码放在这个文件夹：

text

编辑

D:\myproject

进入该目录：

bash

编辑

cd D:\myproject

然后初始化 Git：

bash

编辑

git init

这会在文件夹里生成一个隐藏的 .git 目录，表示这是一个 Git 仓库。

**➕ 四、添加文件并提交（commit）**

**1. 添加所有文件到暂存区**

bash

编辑

git add .

如果只想加某个文件：git add main.py

**2. 提交代码到本地仓库**

bash

编辑

git commit -m "第一次提交：添加主程序和配置文件"

-m 后面是提交说明，务必写清楚做了什么。

**🌐 五、关联远程 GitHub 仓库**

**1. 在 GitHub 上创建一个新仓库**

* 登录 GitHub
* 点击右上角 + → New repository
* 填写仓库名（如 myproject）
* 描述可选
* 选择 **Public（公开）或 Private（私有）**
* ❗ **不要勾选 “Initialize this repository with a README”**（因为我们本地已有代码）
* 点击 Create repository

**2. 关联远程仓库地址**

GitHub 会给你一个仓库地址，类似：

text

编辑

https://github.com/你的用户名/myproject.git

在命令行中运行：

bash

编辑

git remote add origin https://github.com/你的用户名/myproject.git

📌 示例：

bash

编辑

git remote add origin https://github.com/ZhangSan/myproject.git

**🚀 六、推送到 GitHub（第一次推送）**

bash

编辑

git push -u origin main

⚠️ 如果提示分支是 master 而不是 main，可以改成：

bash

编辑

git push -u origin master

首次推送可能需要你登录 GitHub 账号（会弹出浏览器或要求输入用户名密码）。

🔐 注意：GitHub 已不再支持密码登录，推荐使用 **Personal Access Token（PAT）** 替代密码。

**🔐 如何解决“登录失败”？—— 创建 Personal Access Token（重要！）**

**步骤：**

1. 进入 GitHub → 头像 → Settings
2. 左侧菜单 → Developer settings → Personal access tokens → Tokens (classic)
3. 点击 Generate new token → Generate new token (classic)
4. 填写：
   * Note: git-push-token
   * Expiration: 选择过期时间（如 90 天）
   * 权限勾选：repo（全部勾上）
5. 点击 Generate token
6. **复制生成的 token（只显示一次！）**

**使用 token 登录**

当你运行 git push 时，如果提示输入密码，请粘贴这个 token，而不是你的 GitHub 密码。

**🔄 七、后续日常使用流程（记住这 4 步）**

每次修改代码后，只需执行这四步即可同步到 GitHub：

bash

编辑

# 1. 查看修改了哪些文件

git status

# 2. 添加所有更改

git add .

# 3. 提交更改（写清楚做了什么）

git commit -m "添加用户登录功能"

# 4. 推送到 GitHub

git push

✅ 就这么简单！

**🎁 八、额外建议（提升效率）**

**✅ 1. 忽略不需要上传的文件（如缓存、临时文件）**

在项目根目录创建 .gitignore 文件，内容如下：

gitignore

编辑

\_\_pycache\_\_

\*.pyc

.DS\_Store

Thumbs.db

venv/

.env

\*.log

.DS\_Store

这样 Git 就不会上传这些文件。

你可以用这个模板：  
👉 https://github.com/github/gitignore/blob/main/Python.gitignore

**✅ 2. 在 VS Code 中使用 Git（更方便）**

VS Code 内置 Git 支持：

1. 左侧点击「源代码管理」图标
2. 可以查看修改、提交、推送
3. 图形化操作，无需命令行

**✅ 3. 创建 README.md 展示项目**

在项目根目录创建 README.md 文件，用 Markdown 写介绍，例如：

markdown

编辑

# 创建基于 Python 3.11 的虚拟环境

python -m venv myproject-env

# 激活

myproject-env\Scripts\activate

# 此时 pip install 安装的包只属于该项目

GitHub 会自动显示这个文件。

**🎉 总结：一句话学会 Git + GitHub**

**git init → git add . → git commit -m "" → git remote add origin ... → git push**

只要记住这个流程，你就能把任何代码安全地保存到 GitHub！

**🔗 学习资源推荐**

| **类型** | **链接** |
| --- | --- |
| Git 中文文档 | https://git-scm.com/book/zh/v2 |
| GitHub Guides | https://guides.github.com/ |
| 在线练习 Git | https://learngitbranching.js.org/ |

如果你想，我也可以帮你：

1. 自动生成 .gitignore
2. 写一个一键提交脚本
3. 搭建多个项目的统一管理结构

随时告诉我你的需求 😊