**在甲、乙2地用VS CODE寫程式，如何使用GIT，方便程式碼管理?**

我來為您詳細說明如何在兩個地點使用 VS Code 和 Git 進行程式碼管理。

基本設置步驟

**1. 初始環境準備**

在兩台電腦都需要安裝：

# 安裝 Git

# Windows: 下載 Git for Windows

# macOS: brew install git

# Linux: sudo apt install git

# 設定 Git 身份（兩台電腦都要做）

**git config --global user.name "您的姓名"**

**git config --global user.email "您的信箱"**

**2. 建立遠端儲存庫**

選擇一個Git託管服務：

**#GitHub（推薦）：**免費，功能完整

#GitLab：免費私有倉庫

#Bitbucket：小團隊免費

在GitHub建立新repository：

登入GitHub → 點擊"New repository"

輸入專案名稱

選擇公開或私有

點擊"Create repository"

**3. 甲地（第一台電腦）初始化**

# 在專案資料夾中

cd /path/to/your/project

# 初始化 Git 儲存庫

git init

# 添加遠端儲存庫

git remote add origin https://github.com/您的用戶名/專案名.git

# 添加所有檔案

git add .

# 第一次提交

git commit -m "Initial commit"

# 推送到遠端

git push -u origin main

**4. 乙地（第二台電腦）clone專案**

# 複製專案到本地

git clone https://github.com/您的用戶名/專案名.git

# 進入專案資料夾

cd 專案名

日常工作流程：

VS Code 中的 Git 操作

1. 使用 VS Code 內建 Git 功能：

* 左側活動欄點擊「源代碼管理」圖示
* 查看檔案變更狀態
* 暫存檔案（點擊 + 號）
* 寫入提交訊息並提交
* 推送/拉取變更

2. 推薦的 VS Code 擴充功能：

* GitLens — Git supercharged
* Git Graph
* Git History

**標準工作流程**

**甲地工作完成後：**

# 查看狀態

git status

# 添加變更的檔案

git add .

# 或者添加特定檔案

git add filename.py

# 提交變更

git commit -m "描述您做了什麼修改"

# 推送到遠端

git push

**乙地開始工作前：**

# 拉取最新變更

git pull

# 開始工作...

**乙地工作完成後：**

# 同甲地的流程

git add .

git commit -m "描述修改內容"

git push

**分支管理策略**

**功能分支工作流（推薦）**

# 創建新功能分支

git checkout -b feature/新功能名稱

# 在分支上工作和提交

git add .

git commit -m "實作新功能"

# 推送分支到遠端

git push -u origin feature/新功能名稱

# 功能完成後合併到主分支

git checkout main

git pull

git merge feature/新功能名稱

git push

# 刪除已合併的分支

git branch -d feature/新功能名稱

git push origin --delete feature/新功能名稱

**解決衝突**

當兩地同時修改同一檔案時：

# 拉取時出現衝突

git pull

# 會提示有衝突

# 在 VS Code 中打開衝突檔案

# 會看到衝突標記：

# <<<<<<< HEAD

# 您的變更

# =======

# 遠端的變更

# >>>>>>> 分支名

**在 VS Code 中解決衝突：**

1. VS Code會高亮顯示衝突區域
2. 點擊「Accept Current Change」、「Accept Incoming Change」或「Accept Both Changes」
3. 手動編輯合併結果
4. 儲存檔案後提交

**最佳實踐建議**

1. 提交訊息規範

# 好的提交訊息範例

git commit -m "fix: 修復心率計算錯誤"

git commit -m "feat: 新增串口自動偵測功能"

git commit -m "docs: 更新README說明文件"

2. .gitignore 設定

建立 .gitignore 檔案：

# Python

\_\_pycache\_\_/

\*.pyc

\*.pyo

.env

venv/

# VS Code

.vscode/settings.json

# 作業系統

.DS\_Store

Thumbs.db

# 專案特定

config/secret.py

logs/

temp/

3. 定期同步

# 每天開始工作前

git pull

# 工作告一段落就推送

git add .

git commit -m "階段性進度"

git push

4. 備份重要分支

# 為重要版本打標籤

git tag -a v1.0 -m "版本 1.0 發布"

git push –tags

**VS Code 設定建議**

設定檔案 (settings.json)：

json

{

"git.enableSmartCommit": true,

"git.confirmSync": false,

"git.autofetch": true,

"git.autoStash": true

}

緊急情況處理

如果推送失敗：

# 先拉取遠端變更

git pull --rebase

# 然後推送

git push

如果需要回復到之前版本：

# 查看提交歷史

git log --oneline

# 回復到特定提交

git reset --hard 提交ID

這樣的設置可以讓您在甲、乙兩地無縫切換工作，並且保持程式碼的一致性和版本管理。記住要養成「**工作前先拉取，工作後就推送**」的習慣！