**Python + pip + Git 快速指令小抄**

**🐍 Python（版本 / 環境）**

# 檢查版本

python --version

py --version

# 執行程式

python main.py

# 互動模式

python

# 建立虛擬環境 (venv)

py -3.11 -m venv .venv

# 啟用 venv

.\.venv\Scripts\Activate.ps1 # PowerShell

.\.venv\Scripts\activate.bat # CMD

source .venv/bin/activate # macOS/Linux

# 停用 venv

deactivate

**情境**：確保用正確 Python 版本，並讓每個專案獨立管理套件。

**📦 pip（套件管理）**

# 檢查 pip 版本與路徑

pip --version

# 安裝套件

pip install requests

# 安裝指定版本

pip install numpy==1.26.4

# 升級套件

pip install --upgrade pandas

# 列出目前已安裝的套件

pip list

# 尋找某個套件

pip search pyqtgraph

# 匯出目前套件清單（requirements.txt）

pip freeze > requirements.txt

# 依 requirements.txt 安裝

pip install -r requirements.txt

# 移除套件

pip uninstall matplotlib

**情境**：新專案建立 → pip install -r requirements.txt；換電腦 → 直接重建 venv + 安裝。

**🌱 Git（版本控制）**

# 初始化新專案 (在專案資料夾)

git init

# 設定使用者資訊

git config --global user.name "你的名字"

git config --global user.email "你的Email"

# 下載遠端 repo

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

# 檢查目前狀態

git status

# 新增檔案到暫存區

git add filename.py

git add . # 全部檔案

# 提交修改

git commit -m "描述修改內容"

# 查看 commit 歷史

git log --oneline

# 建立並切換新分支

git checkout -b dev

# 推送到遠端

git push origin main

git push origin dev

# 拉取遠端最新

git pull origin main

# 檢查目前 remote 設定

git remote -v

**情境**：

* 第一次交作業/上傳程式：git init → git add . → git commit -m "init" → git push
* 開發新功能：git checkout -b feature-X → 開發 → git commit → git push origin feature-X
* 更新到最新：git pull origin main

**⚡ 常見組合情境**

**1. 建立新專案並安裝套件**

mkdir MyProject

cd MyProject

py -3.11 -m venv .venv

.\.venv\Scripts\Activate.ps1

pip install numpy matplotlib pyqtgraph

pip freeze > requirements.txt

**2. 下載別人的 GitHub 專案並安裝**

git clone https://github.com/xxx/yyy.git

cd yyy

py -3.11 -m venv .venv

.\.venv\Scripts\Activate.ps1

pip install -r requirements.txt

**3. 開發中備份版本**

git add .

git commit -m "完成 BITalino 即時畫圖模組"

git push origin main