04 開發者 / 整合

開發環境建置

📅 最後更新：2026-06-16 | 📌 負責人：KC

本指南幫助開發者建立完整的 MES Gateway 開發環境，從安裝工具到編譯上傳韌體。

01

前置需求

工具	版本	用途
PlatformIO	Core 6.x	韌體編譯與上傳
VS Code	最新版	編輯器（搭配 PlatformIO IDE 外掛）
Node.js	18+	Config Server
Python	3.8+	測試腳本與 Web 上傳工具
Git	最新版	版本控制

02

Step 1：Clone 專案

```bash
git clone <repo-url> PillarArduno
cd PillarArduno
```

03

Step 2：安裝依賴

韌體端

PlatformIO 會在首次編譯時自動安裝 Arduino Core 和 library 依賴（定義在 `platformio.ini`）。

Config Server

```bash
cd config-server
npm install
cd ..
```

Python 工具

```bash
pip install requests pyserial
```

04

Step 3：編譯韌體

```bash
# 編譯（不上傳）— 確認資源使用
pio run

# 確認輸出中的 RAM/Flash 比例在安全範圍：
# RAM:   < 25% 🟢
# Flash: < 85% 🟢
```

05

Step 4：上傳韌體

```bash
# 用 USB 線連接 Opta，確認 port
ls /dev/cu.usbmodem*

# 編譯 + 上傳
pio run -t upload

# 若自動偵測失敗，指定 port
pio run -t upload --upload-port /dev/cu.usbmodemXXXXX
```

> ⚠️ 上傳失敗？嘗試**快速雙擊 Reset 按鈕**進入 DFU 模式。

06

Step 5：上傳 Web UI

> ⚠️ **`upload_web.py` 已廢除**。Web 資產不再單獨上傳：`index.html` ＋核心 `app.js` 編進韌體 PROGMEM（`build_webpage.py` pre-hook），CSS／語系／`workflow.html` 走 QSPI，全部透過 `.mesb` 一檔帶齊。開發機補資產有兩條路：

```bash
# (A) 推薦：灌同版 .mesb (fw+6 資產一次到位，會重燒重開，約 90 秒)
curl -sS -X POST "http://<DEVICE_IP>/api/ota/bundle" \
  -H "Authorization: Bearer smmsadmin" \
  -H "Content-Type: application/octet-stream" \
  --data-binary @release/mes-gateway-v<版本>.mesb

# (B) 純補資產不重燒：逐一 POST web/build/*.gz (先跑 build_webpage.py)
for f in app.js.gz workflow.html.gz styles.css.gz \
         i18n-en.json.gz i18n-zh-TW.json.gz i18n-zh-CN.json.gz; do
  curl -sS -X POST "http://<DEVICE_IP>/api/web?name=$f" \
    -H "Authorization: Bearer smmsadmin" --data-binary @web/build/$f
done
```

> 詳見 `guide-cli-opta.md` §5。

07

Step 6：查看 Serial Log

```bash
# 方式 A：PlatformIO（需要 TTY 環境）
pio device monitor -b 115200

# 方式 B：直接讀取（適用非 TTY 環境，如 Agent 終端機）
ls /dev/cu.usbmodem*
stty -f /dev/cu.usbmodem12201 115200 raw -echo && cat /dev/cu.usbmodem12201

# 方式 C：網頁遠端看（v6.0.1+；設備只能經 VPN、無法插 USB 時）— 需 admin token
curl -s http://<設備IP>/api/log  -H "Authorization: Bearer <admin-token>"   # 純文
curl -s http://<設備IP>/api/diag -H "Authorization: Bearer <admin-token>"   # 開機
```

> 方式 C 也有網頁版：瀏覽器開 `http://<設備IP>/log.html`（或設定頁「📟 設備日誌」鈕），每 5 秒自動刷新 + 頂部橫幅顯示重開原因。log ring 即時吐 RS485 輪詢/MQTT/開機序列；敏感值（密碼/token）已遮罩。**限制**：設備網路死掉（reboot loop）時進不去，改看雲端心跳的 `bootReason` 或現場 USB。

08

Step 7：啟動 Config Server

```bash
cd config-server
npm start
# 預設在 http://localhost:8888
```

09

專案結構速覽

```
PillarArduno/
├── src/                ← 韌體原始碼（32 個 .h/.cpp 檔案）
├── web/                ← Web UI（index.html）
├── config-server/      ← 集中管理伺服器（Node.js）
├── scripts/            ← 工具腳本（上傳 Web、打包 OTA）
├── test/               ← 測試腳本
├── docs/               ← 技術文件（你現在讀的就是）
├── data/               ← 範例資料和配置
└── platformio.ini      ← PlatformIO 專案設定
```

10

常見問題

問題	解法
`pio run` 報錯找不到 board	確認 `platformio.ini` 中 `board = opta`
Serial 無輸出	確認鮑率 115200，重新 Reset 設備
Web UI 空白	QSPI 缺資產 → 灌 `.mesb` 或 POST `web/build/*.gz`（見 Step 5；`upload_web.py` 已廢除）
Config Server 起不來	確認已執行 `npm install`

11

下一步

- 首次部署指南 — 從開箱到上線
- API 參考 — REST API 端點