04 開發者 / 整合

OTA 與內嵌 Web

📅 最後更新：2026-06-16 | 📌 負責人：KC

本文件說明 Arduino Opta MES Gateway 的韌體如何整合 Web UI，以及相關的編譯與 OTA 更新操作流程。

⚠️ **v5.9.155+ 重大更新** 本文件早期版本描述的「全面廢除 `.mesb`、所有前端常駐 PROGMEM」**已部分反轉**。Flash 逼近 100% 上限後，**CSS 與三套語系字典（zh-TW/en/zh-CN）已從 PROGMEM 搬到 QSPI** runtime 服務，且 **`.mesb` 單一檔 OTA 已復活**（一檔帶 firmware + web 資產）。以下內容已更新。

01

1. 架構變更摘要

混合架構（v5.9.155+）：

資源	位置	服務方式
`index.html` + `app.js`（核心）	韌體 PROGMEM（`src/WebPage.h`）	`QSPIWebManager` HTTP GZIP 串流
`styles.css`	QSPI P4 `/cfg/web/`	`serveFile()`（ETag 304 快取）
`i18n-{zh-TW,en,zh-CN}.json`	QSPI P4 `/cfg/web/`	`serveFile()`（ETag 304 快取）
極小英文 fallback 字典	韌體 PROGMEM	QSPI 缺檔時兜底，UI 不全白

- 核心 HTML+JS 仍在 `pio run` 的 pre-build hook（`scripts/build_webpage.py`）被 Gzip + 轉 C byte array 寫入 `src/WebPage.h`；同一腳本**另外**輸出 `web/build/styles.css.gz`、`web/build/i18n-*.json.gz` 供打包進 `.mesb` 投遞到 QSPI。
- **為何要搬**：整包內嵌時 binary 788KB > 780KB OTA brick 上限 → 只能 USB DFU。把 CSS+語系 搬 QSPI 後 binary 跌破 780KB，**OTA 復活**。QSPI/WiFi 並存的 doctrine 也一併被實證推翻（ADR-014）。

02

2. Web UI 開發與建置流程

前端開發者工作流

前端開發者（Vue / Vite 專案）維持原本的開發習慣，唯一的要求是：**將最終編譯出且 Inline 所有 CSS/JS 的單一 `index.html` 檔案，放置或替換掉本專案目錄下的 `web/index.html`**。

韌體編譯者工作流

無需執行任何額外命令，建置與封裝已全自動化！

```bash
# 編譯韌體（不上傳）
pio run -e opta

# 編譯 + 上傳到板子
pio run -e opta -t upload

# 指定 USB port（若自動偵測失敗）
pio run -e opta -t upload --upload-port /dev/cu.usbmodemXXXXX
```

[!IMPORTANT] **務必加上 `-e opta`**。專案的 `platformio.ini` 同時定義了 `opta`（Arduino MCU）與 `native`（本機 Unit Test）兩個環境。若直接執行 `pio run` 不帶 `-e` 參數，PlatformIO 會同時編譯兩個環境，而 `native` 環境因為缺少 `Arduino.h` 標頭檔必然報錯。這個錯誤**不影響韌體編譯結果**，但會讓終端機輸出看起來有紅字 FAILED。加上 `-e opta` 即可避免。

自動化原理：`platformio.ini` 中已加掛 `extra_scripts = pre:scripts/build_webpage.py`。因此在每一次編譯前，腳本會自動檢查 `web/index.html`，並將其壓縮寫入 `src/WebPage.h`，確保韌體時刻包含最新的前端介面。

上傳後驗證

每次上傳後，建議透過 Serial Monitor 確認韌體正常啟動：

```bash
# 先確認 port 名稱
ls /dev/cu.usbmodem*

# 連線監聽開機日誌
stty -f /dev/cu.usbmodem12201 115200 raw -echo && cat /dev/cu.usbmodem12201
```

03

3. OTA 遠端更新操作

📌 **發佈程序的權威 SOP 即本文（見下方各節）**。發佈分兩條程序：「程序 A 一般版本（`.mesb`）」與「程序 B 拓荒包（無版號 `bootstrap .bin`）」，兩者清楚分開。本節只講建置與封裝的技術細節。

因 CSS/語系已搬 QSPI，OTA 需同時帶 firmware + 這些資產，故**使用 `.mesb` 單一檔**（`.bin` 只更新 firmware，不帶資產，僅適用資產未變動或乙太網 raw OTA）。

取得更新檔

`pio run -e opta` 編譯後 firmware 位於 `.pio/build/opta/firmware.bin`；web 資產位於 `web/build/*.gz`（同一次 build 由 `build_webpage.py` 產出）。打包成 `.mesb`：

```bash
python3 scripts/build_ota_bundle.py \
  --fw .pio/build/opta/firmware.bin \
  --asset web/build/styles.css.gz \
  --asset web/build/i18n-en.json.gz \
  --asset web/build/i18n-zh-TW.json.gz \
  --asset web/build/i18n-zh-CN.json.gz \
  --out release/firmware.mesb
python3 scripts/build_ota_bundle.py --verify release/firmware.mesb   # 驗 magic +
```

> **`.mesb` 內含什麼**：韌體 + 6 個網頁資產（`app.js` / `workflow.html` / `styles.css` / `i18n-{en,zh-TW,zh-CN}.json`）+ `favicon.ico`，打包成單一檔。上傳後 firmware → QSPI P2、資產 → P4 `/cfg/web/`，重開即韌體與前端同版本一次到位。

拓荒包（Pioneer）一鍵完整安裝閉環（WI-149）

新設備 / 救磚的端到端開荒流程（已實機驗證）：

1. **USB 首燒**拓荒包 `mes-gateway-bootstrap.bin`（**無版號、固定一顆**；精簡 `.bin`，自包單一開荒頁，不依賴 QSPI 資產。雲端下載連結永不變，詳見本文程序 B）。
2. 開機 → 設備網頁 = **拓荒設定頁**（UID + 網路設定 + 大大的「上傳 .mesb」鈕）。8310 自動過濾 WiFi 欄（型號感知 WI-151）。
3. 在拓荒頁（或完整版）**上傳官方 `.mesb`** → `POST /api/ota/bundle` → `handleMesbUpload` → 設備寫 QSPI + 重開 → **完整版（韌體 + 完整 UI）**。

favicon：拓荒頁內嵌真 `favicon.ico`（base64）；`.mesb` 也打包 `favicon.ico` 給完整版（韌體 `GET /favicon.ico` 由 QSPI `/cfg/web/favicon.ico` 服務）。

ETag 修正（WI-149）：根網頁 ETag 已併入頁面大小（不只 `FW_VERSION`），避免 pioneer → 完整版（同版本號）OTA 後瀏覽器仍顯示舊快取的拓荒頁。

發佈更新

透過 Config Server 上傳 `release/firmware.mesb`（雲端依 `MESB` magic 自動命名 `.mesb`）→ 選設備「部署」。設備依下載 URL 副檔名分流：`.mesb` → `handleMesbUpload`（firmware → QSPI P2 `UPDATE.BIN`、資產 → P4 `/cfg/web/`，含強制 reformat P2 + mbed File API）；`.bin` → `handleOtaUpload`（raw 寫 P2）。詳見使用者手冊 §13。

韌體與前端同版本保證：`.mesb` 一檔內含兩者，OTA 後 QSPI 資產與 firmware 必然同版本，杜絕版本不一致導致 API 溝通失敗。**OTA 寫 QSPI 仍有 soft-brick 風險（可 USB DFU 救回）。**

04

4. 資源佔用與安全水位監控

每次 `pio run -e opta` 完畢時，請務必關注終端機最後輸出的資源使用率：

```
RAM:   [====      ]  42.7% (used 223816 bytes from 523624 bytes)
Flash: [==========]  98.2% (used 772584 bytes from 786432 bytes)   ← v5.9.172 實
```

⚠️ **本專案 Flash 長期貼近上限**：功能持續累積，Flash 已達 ~98%。**真正的硬限制是 `firmware.bin` 必須 < 780000 bytes（OTA partition cap 786432 − overhead），否則 OTA 會被 413 拒絕、只能 USB DFU。** WI-124 把 CSS+語系搬 QSPI 正是為了把 binary 從 788KB 壓回 ~777KB（< 780KB），讓 OTA 復活。**新增功能前務必評估 size，守住 780000 上限。**

資源	🟢 安全	🟡 警戒	🔴 危險
RAM	< 25%	25-40%	> 40%
Flash (binary)	< 740KB	740–780KB	**> 780KB (OTA brick 線)**

若逼近 780KB，優先考慮把更多靜態資源（圖檔、字典）搬 QSPI，而非加大 PROGMEM。