INI_EDITOR 預設模板外部化 ・ 動態預設 ・ 去重

專案:HBSTK / INI_Editor(C:\STK\TOOLS\INI_Editor.exe) 日期:2026-06-05

一、訴求與現象

INI_EDITOR 用來跨工作站編輯 System.ini / stk.ini / stk_agent.ini,核心功能是「補缺 key」「重置為預設」「跨站同步」。原本「預設值模板」是寫死在程式碼(SYSTEM_INI_KEYS 等 dict)裡,衍生三個痛點:

1. 每次新增 key 或調整預設值,都要改程式、重編 exe、重新派送。
2. 預設值每個工作站不一定相同——路徑要跟著工作站(主機 C:\STK、工作站 W:),KMSN 要依站別讀對應 BAT。寫死無法表達。
3. ini 檔會累積重複的 section/key(例如重複的 [Agent] 整段),造成混亂甚至載入異常。

二、踩坑與陷阱

踩坑 / 陷阱	說明
「當前值」不等於「預設值」	一開始想用 build\System.ini 的現值當預設,但那其實是開發機的現值(PRINTER=Epson、DEBUG_LOG=1、ApiKey、開發路徑…),不是出廠預設。硬套會把這些灌進每台客戶端。
S001 vs S002 的差異多是「雜訊」	比對兩站發現大量 [App]/[UI] 差異(KM、StkQtyForm、PPCache…)只是兩台各自設定,不是預設。結論:非路徑 key 一律以 S001(主機)為靜態預設基準。
路徑規則主機/工作站不同	主機 base=C:\STK、工作站 base=W:;LOC_MAIN/LOC_DBF 固定指向 s001(共用主機資料庫);LOC_IMG 為固定共享值。需用「動態規則」表達而非單一字串。
字型 choices 是程式計算的	FontName 等下拉選項由系統字型即時產生,JSON 無法承載 → 外部化時保留程式計算部分,JSON 沒給 choices 就沿用既有。

三、解決方法

做法	內容
預設模板外部化	抽成 `build\ini_template.json`(隨 exe 同層派送)。啟動時`_apply_ini_template_json()` 疊加到既有 schema:覆寫 default、掛動態旗標、補 JSON 才有的新 key。→ 日後加 key / 改預設只改 JSON,免重編。
動態預設解析器	`resolve_default(km, station)` 依旗標算實際預設,補KEY / 重置 / 跨站同步共用,站別取自工作站 ComboBox。
去重 dedupe	`IniFile.dedupe()`:重複 (section,key) 保留第一個、重複 [Section] 標頭併入第一個;載入時自動執行(有重複才標未儲存)。
基準 = S001	靜態 key 以 S001 現值為預設;路徑/KMSN 用動態規則。由 `_gen_ini_template.py` 從 S001+S002 推導生成 JSON。

四、動態規則(JSON 旗標)

旗標	意義	範例(S001 / S002)
default	固定靜態預設	DEBUG_LOG=0、StripSpace=1
dyn=path	路徑:主機 master、工作站 ws	LOC_SYS: C:\STK\s001 / w:\s002
(path 固定 s001)	LOC_MAIN/LOC_DBF 永遠指主機	C:\STK\s001\dbf / w:\s001\dbf
dyn=station	tpl 代入 {ST} 大寫站	PRNASK_HOME_PATH=C:\STK\S001
dyn=station_name	= 工作站代碼	STATION_NAME=S001
dyn=kmsn_bat	讀該站 BAT 的 set kmsn=	主機 GOLOCAL.BAT / 工作站 GO.BAT

五、使用與部署

- 調整預設 / 加 key:編輯 build\ini_template.json 即可,不需重編 exe。
- 重新生成:跑 _gen_ini_template.py 從指定站推導。
- 派送:stk_uploader 的 TOOLS 群組會把 INI_Editor.exe(從 dist)與 ini_template.json(從 build,同源)一併打包,確保 exe 與 json 永遠同步、不漏帶。
- 版本辨識:uploader 視窗標題顯示 build 時間。