

DBF 作業：進度框效能與退出掃描優化

日期 2026-06-06 | 範圍：TLIB/DBF/STK/APPSETUP.PRГ + stk.ini + INI_EDITOR |
ProcessBox 節流 + 動態掃站 + 設定外部化

一、訴求與現象

1. 大量 DBF 讀寫時，一旦顯示進度框（數字型），整個 Disk IO / 資料庫讀寫慢 4-5 倍；把進度框完全拿掉就「飛快」。
2. INDEX / PACK 通知工作站退出時，盲掃固定範圍 S001~（程式為 1 TO 12），對不存在的工作站也照打網路磁碟，太久。客戶端數量不固定。

二、踩坑與陷阱

坑	說明
不是「畫幾次」	進度框從每筆 SHOW → 每 1000 筆 → 只 10 次(0%~100%)，還是慢；整個拿掉才飛快。代表慢的不是重繪「次數」。
同步重繪稅	根因：Harbour 在 DispCount==0 時，每個 @SAY 都「立即同步」flush → Refresh → ProcessMessages → WM_PAINT → ScanAndDrawUda。CJK 畫面每筆一次同步全重繪，被 USRFont 放大，IO 被卡。
ProcessBox 很輕卻拖垮	ProcessBox 非初次只有一行 @SAY 數字，繪圖量極小——證明問題是「同步重繪稅」，不是繪圖量。
盲掃死路徑	BroadcastShutdown 用 FOR 1 TO 12 對每個 W:\Sxxx 寫 FLG/輪詢 File()/Ferase，對不存在的站台在網路磁碟上累積延遲。
8 秒 ACK 逾時	對「資料夾存在、但電腦關機」的站台，因為它有資料夾卻不會回 ACK，仍要空等到 8 秒門檻。

三、解決方法

項目	做法
ProcessBox 時間節流	加 STATIC nLastSec，距上次實際重繪 < 門檻(預設 0.1 秒)就 RETURN 跳過（不 @SAY、不 flush）。重繪最多每秒 ~10 次、與筆數無關 → IO 拿回 4-5 倍、進度看起來反而更順。Abs() 防跨午夜 Seconds() 回繞。

項目	做法
INDEX/PACK 動態掃站	新增 GetActiveStations()：用一次 Directory("W:\S*", "D") 列出 W:\ 下實際存在的 S+3碼 資料夾（每個=一台電腦），BroadcastShutdown / CleanupShutdownFlg 只對存在的站台廣播/清理。有幾台掃幾台、自動適應、不碰不存在的網路路徑。
8 秒 → 可設定	ShutdownAckWait 移到 stk.ini（預設 5）；BroadcastShutdown 讀 Val(GETE("ShutdownAckWait")), <=0 則預設 5。等待/90 秒上限/FLG-ACK 清理邏輯不動。
全面 stk.ini 化	lProcessBox (ProcessBox=Y/N)、節流秒數(ProcessBoxThrottle)、ShutdownAckWait 三者皆 stk.ini 可設定，GETE + 預設值保護，舊版 ini 沒 key 也自動用預設。

四、使用技術與用法

項目	內容
讀取機制	main.prg Main() 先 LoadIni(stk.ini) 把每個 KEY=VALUE 載入成環境變數，之後以 GETE("Key")=AllTrim(GetEnv(Key)) 讀取。STK.PRg 的 PUBLIC 區在 LoadIni 之後執行，可直接 GETE。
stk.ini 新增	[App] 區：ShutdownAckWait=5、ProcessBox=Y、ProcessBoxThrottle=0.1。
客戶端用法	不需重編：改 ProcessBox=N 關進度框、ProcessBoxThrottle=0.2 調節流、ShutdownAckWait=3 縮短等待。
INI_EDITOR	ini_editor.py 的 INI_META（硬編預設模塊）+ 三份 ini_template.json (build / dist / update_pack\TOOLS) 皆同步補上三個 key；因 JSON 疊加到 INI_META，舊版 EXE 只要更新 JSON 即可顯示新 key。
修改檔案	source\TLIB.PRg(ProcessBox)、source\DBF.PRg(BroadcastShutdown+CleanupShutdownFlg+GetActiveStations)、source\STK.PRg + source\APPSETUP.PRg(lProcessBox)、build\stk.ini、build\ini_editor.py、ini_template.jsonx3。