報價單增項無限閃爍修正

日期：2026-05-27　｜　檔案：QUOT.PRG

一、訴求與現象

報價單在使用者特定操作下，畫面紅框區（rows 19-21，品名訊息顯示區）會進入無限空白閃爍狀態，AskPartsBox（零件選單）從未出現，整個 Stk32 卡死，必須強制 KILL 進程才能跳出。

觸發條件對照：

操作流程	結果
進入報價單舊單 → 連續增項 → 連續增項...	正常，不閃
進入報價單舊單 → 增項 → 結束增項（ESC）→ 光棒上移一筆 → 再增項	無限閃爍，必須強制 KILL
同樣操作流程，但改在出貨單（BUYSELL）執行	完全正常，不閃
Clipper 時代（29 年）同樣操作	沒被回報過

二、踩坑與陷阱（兩次失敗嘗試）

一開始誤判為 GTWVT 繪圖問題，前後做了兩次 DispBegin/End 包裝嘗試，皆無效：

嘗試次數	改動位置	想法	結果
1	ChkBotMsg() 整個函式包 DispBegin/End	懷疑 Scroll → PartsMessage → ShowTotal 三步之間 GTWVT 偷 paint	一樣閃
2	ChkBotMsg() ELSE 分支包 DispBegin/End	改鎖定「nClsMethod=2 的兩次 Scroll 不原子」這個更窄的假設	一樣閃

關鍵轉折：

使用者一句「邏輯有問題的話 Clipper
也應該閃」徹底推翻錯誤方向。加上使用者補充「AskPartsBox
根本還沒出場」和「一直閃不會結束 要強制 KILL」，三個事實組合起來，
問題立刻從「繪圖層」轉到「程式邏輯死迴圈」。

三、解決方法

根因：QUOT.PRG `CASE nChoice = 1`（增項）入口缺少 `nLastKey := 0`，導致上方 K_UP/K_DOWN CASE 設定的 nLastKey
殘留，使得增項主迴圈內的 IF 條件永遠走 ELSE
分支，AskPartsBox/AskBarCodeBox 從不呼叫，又無法以 K_ESC 跳出 →
無限迴圈。

關鍵 IF 條件（QUOT.PRG line 761）：

```
IF ( nLastKey = K_INS .AND. AskBarCodeBox(...) ) .OR.
   ( nLastKey = 0     .AND. AskPartsBox(...) )
```

當 nLastKey 殘留 K_UP（或 K_DOWN 等其他鍵碼），兩個條件都成
false，IF 整個失敗 → 走 ELSE → LastKey() 是「1」不是 K_ESC →
LOOP 回頂端 →ChkBotMsg 又清空紅框 → 又走 ELSE → 又 LOOP →
永遠出不來。

修法（加一行）：

```
CASE nChoice = 1
  IF Test(aHead[MAIN_FLAG], ...)
    MessageBox(...)
  ELSE
    nLastKey := 0     // ★ ■■■■
    ShowOrderBody(...)
    DO WHILE .T.
      ...
    ENDDO
  ENDIF
```

四、強力證據：BUYSELL 已有此修正

在診斷過程中發現 BUYSELL.PRG 第 735 行本來就有 `nLastKey := 0`，跟
QUOT.PRG 的唯一結構差異就是這一行：

檔案	CASE nChoice=1 開頭	使用者回報
BUYSELL.PRG	line 735 有 nLastKey := 0	不閃
QUOT.PRG（修正前）	缺 nLastKey := 0	無限閃
QUOT.PRG（修正後）	加 nLastKey := 0	不閃

這項證據解釋了一個關鍵疑問：「為什麼 Clipper 29 年沒事？」答案是 — 這 bug 本來就潛伏在原 Clipper 邏輯中，但 BUYSELL.PRG 早期版本已修補，QUOT.PRG 漏掉而沒人發現。Clipper 時代可能也發生過卡死，但畫面更新慢、使用者習慣 Ctrl+Break 跳出，從未被當「bug」回報。Harbour+GTWVT 下畫面 paint 銳利、閃爍可見性大幅提升，問題才浮上來。

五、使用技術與方法

技術	用途
Python bytes-only 修改 BIG5 PRG	修改檔案時不做 decode/encode 循環，避免中文亂碼；註解直接 .encode("big5")
Harbour CASE/DO WHILE 流程靜態分析	追蹤 nLastKey 在整個 Quot() 函式內的所有寫入點，發現只有 K_UP/K_DOWN/K_PGUP 設定，沒有重置
對比結構相似檔案（QUOT vs BUYSELL）	兩者增項邏輯幾乎相同，唯一差異就是 nLastKey := 0 那一行，直接鎖定 root cause
症狀分類學	「能否自行結束」是關鍵分類：能停 → 繪圖層問題；要 KILL → 邏輯死迴圈
Clipper 對照原則	「Clipper 邏輯一樣 → Harbour 應該也一樣」是強力篩除錯誤假設的判據

六、後續事項

1. BUYSELL.PRG 的 WriteDebugLog 仍保留：
BUYSELL 第 755、760、762、764、775、801 行有 WriteDebugLog("APPEND...") 殘留，應該是之前追問題時加的。確認 QUOT

修正穩定後，可考慮移除以避免每次增項都寫 log。

2. 同類 bug 的預防：
凡是 CASE 入口會「殘留前一個 CASE 的 LOCAL
變數狀態」的設計，都應該在入口明確重置。可考慮全系統 grep 其他 CASE
nChoice=N 是否有類似遺漏。

3. 報價單 ESC 出口擴增（選做）：
QUOT.PRG 只有 K_ESC 能跳出，BUYSELL 還支援 K_F12（line
778）。可考慮統一行為，讓 QUOT 也支援 K_F12 跳出。