

表身翻頁失控根治(PGDN/PGUP)

鍵盤緩衝 backlog + InKey 抽鍵深度之謎;13 張單據、4 次迭代
2026-07-23 HBSTK/source 多檔

一、訴求與現象

比價單(DOMCMP)等單據,表身筆數很多(8000~9000 筆)時用 PGDN 翻頁:連續按住 PGDN 約五秒不放,放開鍵盤後表身仍不受控地一直自己翻頁,直到最後一筆才停。客戶要求:放開鍵盤就要停。

二、根因:鍵盤緩衝 backlog 累積

按住 PGDN 時 OS 以約 30ms 自動重複把 K_PGDN 塞進緩衝;但每頁 ShowBody(8000~9000 筆)渲染要上百 ms——鍵進得比處理得快,緩衝堆積成百上千個。連續翻頁的合併迴圈靠 InKey() 一次消耗一個,放開後這些 backlog 仍被逐一處理 → 一路翻到最後一筆。程式有兩種 idiom:

型態	寫法	陷阱
Shape A	DO WHILE .T. ... IF InKey()=K_PGDN EXIT(inline PageDown)	InKey() 一次消耗一個,放開後排乾 backlog
Shape 1	DO WHILE InKey()=nLastKey + hb_keyPut(nLastKey) + KeyPageUpDown	★hb_keyPut 每圈把鍵「再注入」→進迴圈永遠有鍵→放開仍翻到底

三、偵錯歷程(4 次迭代的懸疑)

這案最耐人尋味的是「同樣加了 hb_keyClear(), INQUIRY 完美、DOMCMP 卻不行」。逐步:

#	當時做法	結果 / 被推翻
1	Shape 1:hb_keyPut(nLastKey) → hb_keyClear()	INQUIRY 放開即停、客戶稱讚。但 Shape A 還沒解
2	Shape A:每頁後加 hb_keyClear()	runaway 沒了,但 DOMCMP 變慢、一頁一頁刷(InKey 零等待在慢渲染後抓不到「還在按」→每頁跳出走完整重畫)
3	Shape A:hb_keyClear() + InKey(0.05)	快了(等 50ms 抓得到還在按),但放開仍殘留 5-6 頁

結果:13 張單據翻頁快、放開即停、不衝到底。客戶實測 INQUIRY / DOMCMP / DOMORD / FORCMP / DOMSCH / STOCK 皆過關。

六、使用技術與心法

技術	用途
hb_keyClear()	清 Harbour 鍵盤緩衝(★不清 Windows 訊息佇列)
DO WHILE InKey() != 0	徹底抽乾 Windows 佇列殘留的整批重複鍵(補 InKey(0.05) 抽不乾的洞)
InKey(0.05)	帶逾時等待,抽乾後判斷有無「全新」重複鍵→分辨仍按住 vs 放開
InKey() 零等待	一次完整抽乾佇列→HB(Shape 1/INQUIRY 靠此一次清光)
hb_keyPut() 反模式	把鍵再注入=製造永不結束的假輸入,務必移除

心法:「連續操作合併迴圈 + 慢渲染」是 backlog runaway

的溫床;放開即停必須把 backlog 徹底抽乾(hb_keyClear

只清一半——Harbour 緩衝,Windows 佇列要 DO WHILE InKey() != 0

才乾淨),再用帶小逾時的 InKey

等「全新」鍵判斷是否續做。★最大教訓:InKey() 零等待與 InKey(nSec) 帶

逾時的「抽鍵深度」不同——結構再像,這個參數差異就能讓一個完美、一個

漏 5-6 頁;debug 這類問題別只看程式結構,要看 InKey

的呼叫型態與相對渲染的時機。