 This is part of **Family API** which allow to create dual-os version of program runs under OS/2 and DOS

Note: This is legacy API call. It is recommended to use 32-bit equivalent

2021/09/17 04:47 · prokushev · 0 Comments
2021/08/20 03:18 · prokushev · 0 Comments

VioPopUp

This call is issued by an application process when it requires a temporary screen to display a momentary message to the user.

Syntax

```
VioPopUp (Options, VioHandle)
```

Parameters

- Options (PUSHORT) - input:Address of the bit flags that indicate which options to the application are being selected.
  - 15-2 - Reserved, set to zero.
  - 1
    - 0 = Non-transparent operation. The video mode is set to text-mode 3, 3*, 3+, 7, or 7+. The highest resolution supported by the primary display adapter configured in the system is selected. The screen is cleared, and the cursor is positioned in the upper left corner of the display.
    - 1 = Transparent operation. If the video mode of the outgoing foreground session is text (mode 2, 3, 7, or one of the * or + variations of these modes), no mode change occurs. The screen is not cleared, and the cursor remains in its current position. If transparent operation is selected, and if the video mode of the outgoing foreground session is not text (or if the outgoing foreground session has a VioSavRedrawWait thread), the pop-up request is refused. A unique error code ERROR_VIO_TRANSPARENT_POPUP is returned in this case.

OS/2 is responsible for saving and restoring the physical display buffer of the previous foreground session around a pop-up. This is true whether transparent or non-transparent operation is selected.

- 0
  - 0 = Return with unique error code ERROR_VIO_EXISTING_POPUP if pop-up is not immediately available.
  - 1 = Wait if pop-up is not immediately available.
- VioHandle (HVIO) - input : Reserved words of 0s.

Return Code

rc (USHORT) - return:Return code descriptions are:

- 0 NO_ERROR
- 405 ERROR_VIO_NO_POPUP
- 406 ERROR_VIO_EXISTING_POPUP
- 483 ERROR_VIO_TRANSPARENT_POPUP

Remarks

VioPopUp is normally issued by the application when it is running in the background and wishes to immediately display a message to the user without waiting to become the active foreground session.

When an application process issues VioPopUp, it should wait for the return from the request. If the process allows any of its threads to write to the screen before VioPopUp returns a successful return code, the screen output may be directed to the application's normal video buffer rather than to the pop-up screen. If the process allows any of its threads to issue keyboard or mouse calls before VioPopUp returns a successful return code, the input is directed from the application's normal session. Once the process that issued VioPopUp receives a successful return code, video and keyboard calls issued by any of the threads in the pop-up process are directed to the pop-up screen. This continues until the process issues VioEndPopUp. At that time video and keyboard calls resume being directed to the application's normal video buffer.

There may be only one pop-up in existence at any time. If a process requests a pop-up and a pop-up already exists, the process has the choice of waiting for the prior pop-up to complete or receiving an immediate return with an error code. The error code indicates that the operation failed due to an existing pop-up having captured the screen.

Video pop-ups provide a mechanism for a background application to notify the operator of an abnormal event the operator must take some action. When considering the suitability of using pop-ups in a particular situation, the possible disruptive effect of pop-ups to the operator should be considered. If the operator were interrupted frequently by pop-ups issued by background applications, the operator would not effectively work with the foreground application.

While a video pop-up is in the foreground, the operator cannot use the hot key to switch to another application or to the shell. Before the operator can switch another application or the shell to the foreground, the pop-up application must issue VioEndPopUp.

While a video pop-up is in effect, all video calls from the previous foreground session are blocked until the process that issued VioPopUp issues VioEndPopUp.

When VioPopUp is issued, only the process within the session that issued VioPopUp is brought to the foreground. Assuming the session was already the foreground session, any video calls issued by other processes in that session are blocked until the process that issued VioPopUp issues VioEndPopUp.

DosExecPgm may not be issued by a process during a pop-up. The following video calls are the only calls that may be issued during the pop-up by the process that issued VioPopUp:

- VioEndPopUp

- VioScrollLf
- VioGetConfig
- VioSetCurPos
- VioGetCp
- VioSetCurType
- VioGetFont
- VioSetCp
- VioGetAnsi
- VioSetFont
- VioGetState
- VioSetState
- VioGetCurPos
- VioWrtNChar
- VioGetCurType
- VioWrtNAttr
- VioGetMode
- VioWrtNCell
- VioReadCharStr
- VioWrtCharStr
- VioReadCellStr
- VioWrtCharStrAtt
- VioScrollRt
- VioWrtCellStr
- VioScrollUp
- VioWrtTTY
- VioScrollDn

Selectors to the physical display buffer that the issuing process obtained on a prior VioGetPhysBuf call may not be used during the pop-up.

When an application registers a replacement for VioPopUp within a session, the registered routine is invoked only when that session is in the foreground. If VioPopUp is issued when that session is in the background, the OS/2 default routine is invoked. If the application's session is using a keyboard or mouse monitor, the monitor does not intercept data while the pop-up is active.

PM Considerations

This function can be used from within a PM application. Kbdxxx, Mouxxx, and Vioxxx calls (with a zero handle) are all allowed between VioPopUp and VioEndPopUp, and are directed to the pop-up screen. An error is returned if issued with a non-zero handle.

Bindings

C

```
#define INCL_VIO
```

```
USHORT  rc = VioPopUp(Options, VioHandle);

PUSHORT Options;        /* Option Flags */
HVIO    VioHandle;      /* Vio handle */

USHORT  rc;             /* return code */
```

MASM

```
EXTRN   VioPopUp:FAR
INCL_VIO            EQU 1

PUSH@  WORD     Options        ;Option Flags
PUSH   WORD     VioHandle      ;Vio handle
CALL   VioPopUp

Returns WORD
```

http://www.edm2.com/index.php/VioPopUp_(OS/2_1.x)

Family API		
DOS	Process Manager	DosBeep DosExit DosSleep DosExecPgm
	File Manager	DosChDir DosChgFilePtr DosClose DosDelete DosDupHandle DosMkDir DosMove DosQCurDir DosQCurDisk DosSetFileMode DosOpen DosQFileInfo DosRead DosQFileMode DosQFSInfo DosQVerify DosRmDir DosSelectDisk DosFindClose DosFindFirst DosFindNext DosSetFileInfo DosSetVerify DosWrite DosFileLocks DosSetFHandState DosNewSize DosBufReset DosQFHandState DosSetFSinfo DosShutdown
	Memory Manager	DosFreeSeg DosSubAlloc DosSubFree DosSubSet DosAllocHuge DosAllocSeg DosReallocHuge DosReallocSeg DosGetHugeShift DosCreateCSAlias
	NLS	DosCaseMap DosGetCtryInfo DosGetDBCSEv DosSetCtryCode DosGetCollate DosGetMessage DosInsMessage DosPutMessage
	Date and Time	DosSetDateTime DosGetDateTime
	Devices	DosDevConfig DosDevIOCtl DosDevIOCtl2
	Signals	DosHoldSignal DosSetSigHandler
	Misc	BadDynLink DosGetEnv DosGetMachineMode DosGetVersion DosError DosErrClass DosSetVec
KBD		KbdCharIn KbdFlushBuffer KbdGetStatus KbdSetStatus KbdStringIn KbdPeek
VIO		VioGetBuf VioGetConfig VioGetCurPos VioGetCurType VioGetPhysBuf VioReadCellStr VioReadCharStr VioScrollUp VioScrollDn VioScrollLf VioScrollRt VioScrUnLock VioSetCurPos VioSetCurType VioSetMode VioGetMode VioShowBuf VioWrtCellStr VioWrtCharStr VioWrtCharStrAtt VioWrtNAttr VioWrtNCell VioWrtNChar VioWrtTTY VioScrLock VioPopUp
Tools		BIND
Modules		DOSCALLS.DLL VIOCALLS.DLL KBDCALLS.DLL MSG.DLL
Libraries		API.LIB OS2386.LIB FAPI.LIB DOSCALLS.LIB SUBCALLS.LIB

2018/08/25 15:05 · prokushev · 0 Comments