



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](#)

VioSetState

This call performs one of the following functions; set palette registers, sets the overscan (border) colour, set the blink/background intensity switch, set color registers, set the underline location, or set the target VioSetMode display configuration.

Syntax

```
VioSetState (RequestBlock, VioHandle)
```

Parameters

- RequestBlock (PVOID) - input : Address of the video state structures consisting of six different structures depending on the request type:
 - 0 - Set palette registers
 - 1 - Set overscan (border) color
 - 2 - Set blink/background intensity switch
 - 3 - Set color registers
 - 4 - Reserved
 - 5 - Set underline location
 - 6 - Set target VioSetMode display configuration
 - 7 - Reserved

The six structures, depending on request type, are:

	Applies to	length (USHORT) - input : Length of structure, including length.	reqtype (USHORT) - input	
VIOPALSTATE	EGA, VGA, or IBM Personal System/2 Display Adapter	38 - Maximum valid value	Request type 0 for palette registers	palette (USHORT) - input: First palette register in the palette register sequence; must be specified in the range 0 through 15. The palette registers are returned in sequential order. The number returned is based upon length.
				color (USHORT*(length-6)/2) - input : Color value for each palette register. The maximum number of entries in the color value array is 16
VIOOVERSCAN	CGA, VGA, or IBM Personal System/2 Display Adapter	Only valid value.	Request type 1 for overscan (border) color.	Color value.
VIOINTENSITY	CGA, EGA, MCGA, VGA, or IBM Personal System/2 Display Adapter	Only valid value.	Request type 2 for blink/background intensity switch.	switch (USHORT) - input : Switch set as:
				0 - Blinking foreground colors enabled. 1 - High intensity background colors enabled.
VIOCOLORREG	VGA, or IBM Personal System/2 Display Adapter	12 - Only valid value.	Request type 3 for color registers.	first color (USHORT) - input : First color register to set in the color register sequence; must be specified in the range 0 through 255. The color registers are set in sequential order.
				number color (USHORT) - input : Number of color registers to set; must be specified in the range 1 through 256.
				dataarea (PCH) - input : Far address of a data area containing one three-byte entry for each color register to be set. The format of each entry is as follows:
				Byte 1 - Red value
				Byte 2 - Green value
				Byte 3 - Blue value

	Applies to	length (USHORT) - input : Length of structure, including length.	reqtype (USHORT) - input	
VIOSETULINELOC	EGA, VGA, or IBM Personal System/2 Display Adapter	Only valid value.	Request type 5 to set the scan line for underlining. Underlining is enabled only when the foreground color is 1 or 9.	scanline (USHORT) - input : Scan line minus 1. Values of 0 through 31 are acceptable. A value of 32 means underlining is disabled.
VIOSETTARGET		Only valid value.	Request type 6 to set display configuration to be the target of the next VioSetMode.	:select (USHORT) - input : Configuration: 0 - Default selection algorithm. See VioSetMode. 1 - Primary 2 - Secondary

- VioHandle (HVIO) - input : Reserved word of 0s.

Return Code

;rc (USHORT) - return:Return code descriptions are: *0 NO_ERROR *355 ERROR_VIO_MODE *421 ERROR_VIO_INVALID_PARMS *436 ERROR_VIO_INVALID_HANDLE *438 ERROR_VIO_INVALID_LENGTH *465 ERROR_VIO_DETACHED *494 ERROR_VIO_EXTENDED_SG

Family API Considerations

Request type = 6, Set Target VioSetMode Display Configuration, and request type = 5, Set Underline Location, are not supported in the family API.

Some options operate differently in the DOS mode than in the OS/2 mode. Therefore, the following considerations applies to VioSetMode when coding for the DOS mode: * VioSetMode clears the screen.

Bindings

C

<PRE> typedef struct _VIOPALSTATE {

```
USHORT  cb;                /* Length of this structure in bytes */
USHORT  type;             /* Request type=0 get palette registers */
USHORT  iFirst;          /* First palette register to return */
USHORT  acolor[1];       /* Color value palette register */
```

```
}VIOPALSTATE;
```

```
typedef VIOPALSTATE far *PVIOPALSTATE;
```

```
typedef struct _VIOOVERSCAN {
```

```
USHORT  cb;                /* Length of this structure */  
USHORT  type;              /* Request type=1 get overscan  
                           (border) color */  
USHORT  color;             /* Color value */  
}VIOOVERSCAN;
```

```
typedef VIOOVERSCAN far *PVIOOVERSCAN;
```

```
typedef struct _VIOINTENSITY {
```

```
USHORT  cb;                /* Length of this structure */  
USHORT  type;              /* Request type=2 get blink/background  
                           intensity switch */  
USHORT  fs;                /* Value of blink/background switch */  
}VIOINTENSITY;
```

```
typedef VIOINTENSITY far *PVIOINTENSITY;
```

```
typedef struct _VIOCOLORREG { /* viocreg */
```

```
USHORT  cb;                /* Length of this structure */  
USHORT  type;              /* Request type=3 get colorregs */  
USHORT  firstcolorreg;     /* First color register */  
USHORT  numcolorregs;      /* Number of color registers */  
PCH     colorregaddr;      /* Address of color registers */  
}VIOCOLORREG;
```

```
typedef VIOCOLORREG far *PVIOCOLORREG;
```

```
typedef struct _VIOSETLINELOC { /* violine */
```

```
USHORT  cb;                /* Length of this structure */  
USHORT  type;              /* Request type=4 get scanline */  
USHORT  scanline;          /* Scanline number */  
}VIOSETLINELOC;
```

```
typedef VIOSETLINELOC far *PVIOSETLINELOC;
```

```
typedef struct _VIOSETTARGET { /* viosett */
```

```
USHORT  cb;                /* Length of this structure */  
USHORT  type;              /* Request type=5 get default algorithm */  
USHORT  defaultalgorithm;  /* Default algorithm */  
}VIOSETTARGET;
```

```
typedef VIOSETTARGET far *PVIOSETTARGET;
```

```
#define INCL_VIO
```

```
USHORT rc = VioSetState(RequestBlock, VioHandle);
```

```
PVOID RequestBlock; /* Request block */ HVIO VioHandle; /* Video handle */
```

```
USHORT rc; /* return code */ </PRE>
```

MASM

```
<PRE> VIOPALSTATE struc
```

```
viopal_cb          dw ? ;Length of this structure in bytes
viopal_type        dw ? ;Request type=0 get palette registers
viopal_iFirst      dw ? ;First palette register to return
viopal_acolor      dw 1 dup (?) ;Color value palette register
```

```
VIOPALSTATE ends
```

```
VIOOVERSCAN struc
```

```
vioos_cb          dw ? ;Length of this structure
vioos_type         dw ? ;Request type=1 get overscan (border) color
vioos_color        dw ? ;Color value
```

```
VIOOVERSCAN ends
```

```
VIOINTENSITY struc
```

```
vioint_cb         dw ? ;Length of this structure
vioint_type        dw ? ;Request type=2 get blink/background
                   ; intensity switch
vioint_fs          dw ? ;Value of blink/background switch
```

```
VIOINTENSITY ends
```

```
VIOCOLORREG struc
```

```
viocreg_cb        dw ? ;
viocreg_type       dw ? ;
viocreg_firstcolorreg dw ? ;
viocreg_numcolorregs dw ? ;
viocreg_colorregaddr dd ? ;
```

```
VIOCOLORREG ends
```

```
VIOSETULINELOC struc
```

```
viouline_cb          dw ? ;  
viouline_type       dw ? ;  
viouline_scanline   dw ? ;
```

VIOSETULINELOC ends

VIOSETTARGET struc

```
viosett_cb          dw ? ;  
viosett_type       dw ? ;  
viosett_defaultalgorithm dw ? ;
```

VIOSETTARGET ends

EXTRN VioSetState:FAR INCL_VIO EQU 1

PUSH@ OTHER RequestBlock ;Request block PUSH WORD VioHandle ;Video handle CALL VioSetState

Returns WORD </PRE>

[http://www.edm2.com/index.php/VioSetState_\(OS/2_1.x\)](http://www.edm2.com/index.php/VioSetState_(OS/2_1.x))

From:
<https://ftp.osfree.org/doku/> - **osFree wiki**

Permanent link:
<https://ftp.osfree.org/doku/doku.php?id=en:docs:fapi:vioresetstate&rev=1634199907>

Last update: **2021/10/14 08:25**

