

[Articles](#) > [Assembleur](#) > [Graphismes](#) > [Draw](#) > [Comment tracer un point ?](#) >

# Comment tracer un point ?

Mis à jour le 24 jan 2008 18:31:19

## Description

écriture d'un pixel en assembleur x86

## Sommaire

Comment tracer un point ?

écriture d'un pixel en assembleur x86

# Comment tracer un point ?

## Comment tracer un point ?

### Composition d'une image

Chaque image est composée de lignes et de colonnes.

Dans la mémoire, une image est constituée d'une série de lignes.

Chaque pixel est codé sur 4 octets selon le format ARGB :

A (alpha) : utilisé pour la transparence

R (red) : composante rouge

G (green) : composante verte

B (blue) : composante bleue

**EXEMPLE :**

Les pixels composant une image de 3 lignes et 5 colonnes se présentent ainsi en mémoire :

```
L0C0 L0C1 L0C2 L0C3 L0C4 L1C0 L1C1 L1C2 L1C3 L1C4 L2C0 L2C1 L2C2 L2C3
L2C4
```

avec :

L = ligne

C = colonne

## Adresse d'un pixel

L'adresse d'un pixel est donnée par la formule suivante :

$$\text{Adresse} = \text{lpSurface} + (y * \text{dwWidth} + x) \ll 2$$

avec  $0 \leq x < \text{dwWidth}$  ;  $0 \leq y < \text{dwHeight}$

## Dessiner un pixel

### prototype de la fonction

```
drawPixel PROTO near C, ddsd:DWORD, x:DWORD, y:DWORD, color:DWORD
```

### description

Pour dessiner un pixel sur une image, on vérifie d'abord q'un descripteur d'image a été fourni.

On vérifie également si le pixel dont les coordonnées sont transmises appartient à l'image.

Si tous les paramètres sont OK, on dessine le pixel sur l'image de la manière suivante : on calcule d'abord l'adresse du pixel à l'aide de la formule précédente puis on écrit la couleur du pixel à cette adresse.

### code de la fonction

```
drawPixel PROC near C, ddsd:DWORD, x:DWORD, y:DWORD, color:DWORD
    pushad
    ; if(!ddsd) return
    mov esi
,ddsd and esi,esi
    jz
    fin ; if(y<0) return
```

```

    mov eax
,y and eax,eax
    js
fin ; if(y>=height) return
    mov ecx,(drawDesc ptr [esi])
.dwHeight cmp eax,ecx
    jge
fin ; if(x<0) return
    mov ebx
,x and ebx,ebx
    js
fin ; if(x>=width) return
    mov edx,(drawDesc ptr [esi])
.dwWidth cmp ebx,edx
    jge
fin ; ---PutPixel---
    mov edi,(drawDesc ptr [esi])
.lpSurface mul edx
    add eax,ebx
    shl eax,2
    add edi,eax
    mov eax
,color mov dword ptr [edi],eax
    fin: popad
    ret
drawPixel ENDP

```