# Références SDL2

# pour eForth Windows

**version 1.2 - mercredi 6 novembre 2024**


## Auteur


- Marc PETREMANN

# Contents

# SDL2

## CreateRenderer window index flag -- render

Créer un contexte de rendu 2D pour une fenêtre.

Paramètres :

- **window** la fenêtre où le rendu est affiché
- **index** l'index du pilote de rendu à initialiser, ou -1 pour initialiser le premier prenant en charge les indicateurs demandés
- flags 0, ou un ou plusieurs SDL_RendererFlags combinés par OU.

```
variable WIN0

z" My first window with SDL2"
    X0_SCREEN_POSITION Y0_SCREEN_POSITION SCREEN_WIDTH SCREEN_HEIGHT
    SDL_WINDOW_SHOWN   SDL.CreateWindow  WIN0 !

variable REN0

WIN0 @ -1 0 CreateRenderer  REN0 !
```

## CreateTextureFromSurface render surface -- 0|texture

Créer une texture à partir d'une surface existante.

## CreateWindow zstr x y w h fl -- win

Créez une fenêtre avec la position, les dimensions et les indicateurs spécifiés.

Paramètres :

- **title** le titre de la fenêtre, en codage UTF-8
- **x** la position x de la fenêtre, SDL_WINDOWPOS_CENTERED ou SDL_WINDOWPOS_UNDEFINED
- **y** la position y de la fenêtre, SDL_WINDOWPOS_CENTERED ou SDL_WINDOWPOS_UNDEFINED
- **w** la largeur de la fenêtre, en coordonnées d'écran
- **h** la hauteur de la fenêtre, en coordonnées d'écran
- **flags** 0, ou un ou plusieurs SDL_WindowFlags combinés par OU

Renvoie **win** qui a été créé ou 0 en cas d'échec.

```
\ define size and position for SDL window
800 constant SCREEN_WIDTH
400 constant SCREEN_HEIGHT
200 constant X0_SCREEN_POSITION
 50 constant Y0_SCREEN_POSITION

z" My first window with SDL2"
    X0_SCREEN_POSITION Y0_SCREEN_POSITION
    SCREEN_WIDTH SCREEN_HEIGHT
    SDL_WINDOW_SHOWN CreateWindow
    value WIN0
```

## Delay ms -- fl

Attend un nombre spécifié de millisecondes.

## DestroyRenderer render -- fl

Détruit le contexte de rendu d'une fenêtre et libère les textures associées.

```
\ free ressources, end renderer and window
: freeRessources ( -- )
    REN0 DestroyRenderer drop
    WIN0 DestroyWindow   drop
    Quit
  ;
```

## DestroyWindow win -- fl

Détruit une fenêtre

```
\ WIN0 must be declared by value and set by CreateWindow
WIN0 DestroyWindow
```

## GetError -- n

Récupére un message sur la dernière erreur survenue sur le fil actuel.

## GetNumRenderDrivers -- n|err

Obtient le nombre de pilotes de rendu 2D disponibles pour l'affichage actuel.

Renvoie un nombre >= 0 en cas de succès ou un code d'erreur négatif en cas d'échec.

## GetTicks -- ms

Obtient le nombre de millisecondes depuis l'initialisation de la bibliothèque SDL.

## GetWindowFlags window -- win-flag

Récupère les flags de fenêtre.

## GetWindowSize windows *w *h -- fl

Obtient la taille de la zone client d'une fenêtre.

## GetWindowSizeInPixels windows *w *h -- fl

Récupère la taille d'une fenêtre en pixels.

Paramètres :

- **window** la fenêtre à partir de laquelle la taille du dessin doit être interrogée
- **w** un pointeur vers une variable pour stocker la largeur en pixels
- **h** un pointeur vers une variable pour stocker la hauteur en pixels

```
variable WIN.width
variable WIN.height
: draw ( -- )
    REN0 255 255 255 255 SetRenderDrawColor drop
    REN0 RenderClear drop
    REN0 RenderPresent drop
    WIN0 WIN.width WIN.height GetWindowSizeInPixels drop
  ;
draw
WIN.width  @ .    \ display: 800
WIN.height @ .    \ display: 400
```

## HideWindow window -- fl

Cache la fenêtre.

## Init n -- n

Initialise la librairie SDL.

n doit être une valeur:

SDL_INIT_TIMER \ timer subsystem

SDL_INIT_AUDIO \ audio subsystem

SDL_INIT_VIDEO \ video subsystem; automatically initializes the events subsystem

SDL_INIT_JOYSTICK \ joystick subsystem; automatically initializes the events subsystem

SDL_INIT_HAPTIC \ haptic (force feedback) subsystem

SDL_INIT_GAMECONTROLLER \ controller subsystem; automatically initializes the joystick subsystem

SDL_INIT_EVENTS \ events subsystem

SDL_INIT_SENSOR

Renvoie 0 en cas de succès ou un code d'erreur négatif en cas d'échec. Appeler `GetError` pour plus d'informations.

```
\ Initialize SDL with error management
: SDL.init ( n -- )
    Init
    if
        ." SDl could not intialize! SDL_Error: " getError .
    then
  ;
SDL_INIT_VIDEO SDL.init
```

## LoadBMP file -- surface

Charger une surface à partir d'un fichier.

Ne pas utiliser ce mot directement, mais `SDL.load-image`.

```
: LoadBMP  ( zstr -- 0|surface )
    z" rb" RWFromFile 1 LoadBMP_RW
  ;

: SDL.load-image  ( zstr -- surface )
    LoadBMP ?dup 0= if
        drop -1 SDL.error
    then
  ;
```

## LoadBMP_RW RWops freesrc -- 0|surface

Charge une image BMP à partir d'un flux de données SDL recherchable.

Paramètres :

- **RWops** le flux de données pour la surface
- **freesrc** différent de zéro pour fermer le flux après avoir été lu.

Renvoie un pointeur vers une nouvelle structure SDL_Surface ou NULL en cas d'erreur.

## MaximizeWindow window -- fl

Fait une fenêtre aussi grande que possible.

## MinimizeWindow windows -- fl

Réduit une fenêtre vers la barre des tâches sous forme d'icone.

## Quit --

Nettoie tous les sous-systèmes initialisés.

## RaiseWindow window -- fl

Élève une fenêtre au-dessus des autres fenêtres et définit le focus d'entrée.

## RenderClear render -- 0|err

Efface la cible de rendu actuelle avec la couleur de dessin.

```
\ REN0 is a value previsously initialized with CreateRenderer
REN0 RenderClear drop
```

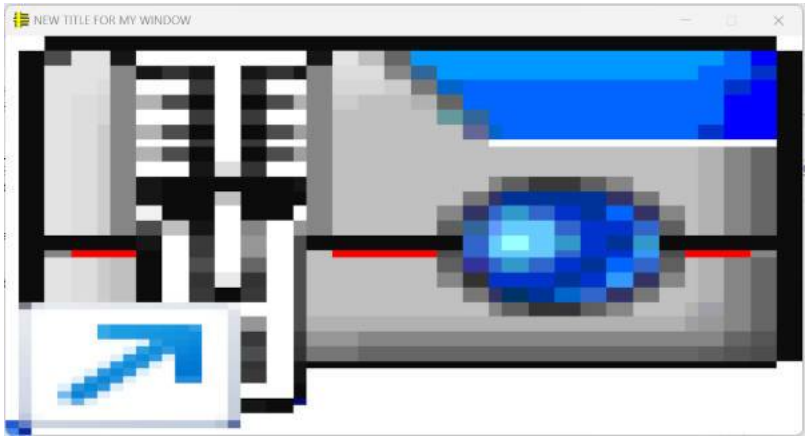
## RenderCopy render texture srcrect dstrect -- 0|err

Copie une partie de la texture sur la cible de rendu actuelle.

Paramètres :

- **renderer** le contexte de rendu.
- **texture** la texture source.
- **srcrect** la structure SDL_Rect source ou NULL pour la texture entière.
- **dstrect** la structure SDL_Rect de destination ou NULL pour la cible de rendu entière ; la texture sera étirée pour remplir le rectangle donné.

```
z" SDL2/logo01.bmp" constant ZFILE
0 value IMAGE01     \ is surface
0 value TEXTURE01
\ load an image
ZFILE SDL.load-image to IMAGE01
REN0 IMAGE01 CreateTextureFromSurface to TEXTURE01
REN0 TEXTURE01 NULL NULL RenderCopy drop
REN0 RenderPresent drop
```




## RenderCopyEx render texture srcrect dstrect angle center flip -- 0|err

Copie une partie de la texture dans le rendu actuel, avec rotation et retournement facultatifs.

Paramètres

- **renderer** le contexte de rendu.
- **texture** la texture source.
- **srcrect** la structure SDL_Rect source ou NULL pour la texture entière.
- **dstrect** la structure SDL_Rect de destination ou NULL pour la cible de rendu entière.
- **angle** un angle en degrés qui indique la rotation qui sera appliquée à dstrect, en le faisant pivoter dans le sens des aiguilles d'une montre.
- **center** un pointeur vers un point indiquant le point autour duquel dstrect sera tourné (si NULL, la rotation sera effectuée autour de dstrect.w / 2, dstrect.h / 2).
- **flip** une valeur SDL_RendererFlip indiquant les actions de retournement à effectuer sur le texture.

## RenderDrawLine render x0 y0 x1 y1 -- 0|err

Trace une ligne sur la cible de rendu actuelle.

Paramètres :

- **renderer** le contexte de rendu
- **x1** la coordonnée x du point de départ
- **y1** la coordonnée y du point de départ
- **x2** la coordonnée x du point final

- **y2** la coordonnée y du point final

Renvoie 0 en cas de réussite ou un code d'erreur négatif en cas d'échec.

```
\ color to white - draw simple line
REN0 255 255 255 255 SetRenderDrawColor drop
REN0 10 20 1200 45 RenderDrawLine drop
REN0 1200 45 10 100 RenderDrawLine drop
REN0 RenderPresent drop
```

## RenderDrawLines render *points count -- 0|err

Dessine une série de lignes connectées sur la cible de rendu actuelle.

```
6 constant STAR_COUNT
create STAR_POINTS
     20 L,  20 L,    150 L,  15 L,
     15 L, 140 L,     80 L,  10 L,
    130 L, 130 L,     20 L,  20 L,
REN0 STAR_POINTS STAR_COUNT RenderDrawLines drop
REN0 RenderPresent drop
```


## RenderDrawRect render *rect --

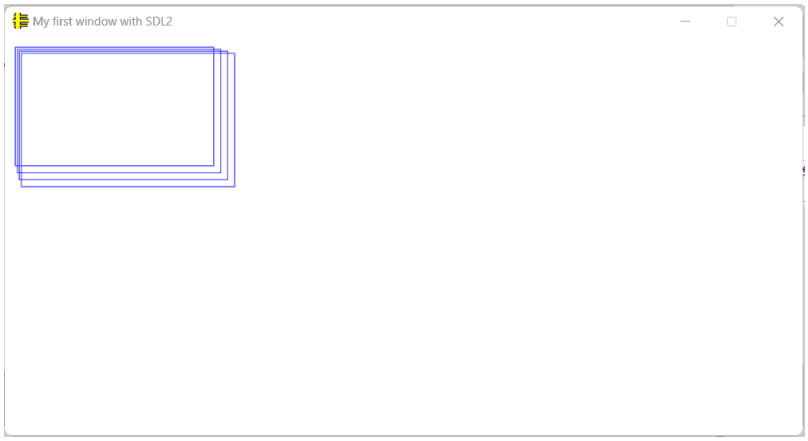
Dessine un rectangle dans le rendu actuel.

```
create RECT
     10 L,   10 L,  200 L,  120 L,

REN0 63 63 255 255 SetRenderDrawColor drop
REN0 RECTS RenderDrawRect drop
REN0 RenderPresent drop
```

## RenderDrawRects render *rect count

Dessine un certain nombre de rectangles sur le rendu actuel.

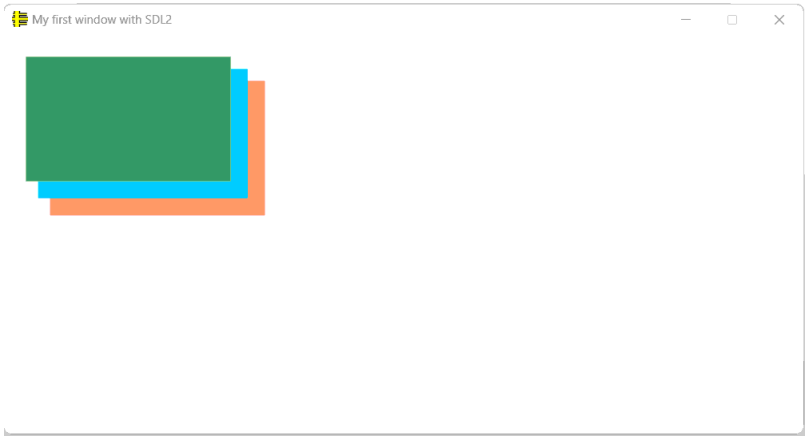
```
4 constant RECT_COUNT
create RECTS
     10 L,   10 L,  200 L,  120 L,
     12 L,   12 L,  205 L,  125 L,
     14 L,   14 L,  210 L,  130 L,
     16 L,   16 L,  215 L,  135 L,

REN0 63 63 255 255 SetRenderDrawColor drop
REN0 RECTS RECT_COUNT RenderDrawRects drop
REN0 RenderPresent drop
```




## RenderFillRect render *rect -- 0|err

Remplit un rectangle dans le rendu actuel avec la couleur de dessin.

```
create RECTS
     10 L,   10 L,  200 L,  120 L,
     22 L,   22 L,  205 L,  125 L,
     34 L,   34 L,  210 L,  130 L,
     46 L,   46 L,  215 L,  135 L,

REN0 $FF $7D $5A $FF SetRenderDrawColor drop
REN0 3 RECTS ->SDL_Rect RenderFillRect drop

REN0 $45 $DC $FF $FF SetRenderDrawColor drop
REN0 2 RECTS ->SDL_Rect RenderFillRect drop

REN0 $45 $A3 $5E $FF SetRenderDrawColor drop
REN0 1 RECTS ->SDL_Rect RenderFillRect drop

REN0 RenderPresent drop
```




## RenderGetLogicalSize render *width *height -- 0|err

Obtient une résolution indépendante de l'appareil pour le rendu.

Ne fonctionne que si `RenderSetLogicalSize` a été utilisé. Sinon stocke 0.

```
variable REN0.width
variable REN0.height

REN0 REN0.width REN0.height RenderGetLogicalSize drop
```

## RenderPresent render --

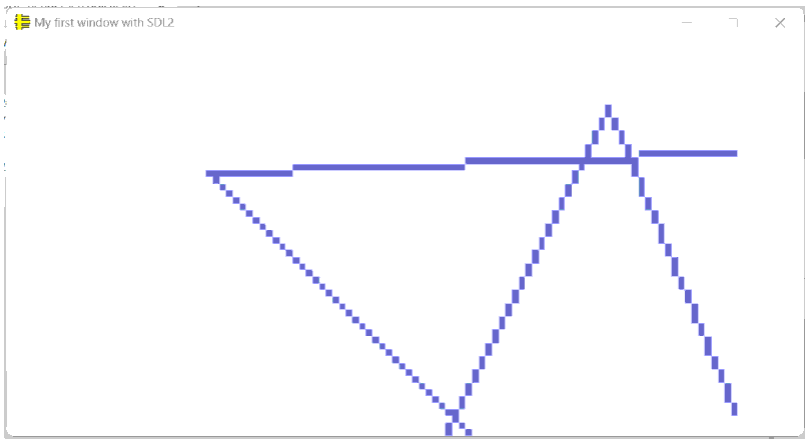
Mettre à jour l'écran avec tout rendu effectué depuis l'appel précédent.

## RenderSetLogicalSize render w h -- 0|err

Définit une résolution indépendante du périphérique pour le rendu.

Cette fonction utilise la fenêtre d'affichage et la fonctionnalité de mise à l'échelle pour permettre une résolution logique fixe pour le rendu, quelle que soit la résolution de sortie réelle. Si la résolution de sortie réelle n'a pas le même rapport hauteur/largeur, le rendu de sortie sera centré dans l'affichage de sortie.

```
REN0 100 60 RenderSetLogicalSize drop
```




## RestoreWindow window -- fl

Restaurer la taille et la position d'une fenêtre minimisée ou maximisée.

## RWFromFile file mode -- 0|RWops

Utilisez ce mot pour créer une nouvelle structure SDL_RWops pour la lecture et/ou l'écriture dans un fichier nommé.

```
: LoadBMP  ( zstr -- 0|surface )
    z" rb" RWFromFile 1 LoadBMP_RW
  ;
```

## SDL2.dll -- <name>

point d'entrée vers la librairie SDL2.dll

```
\ Destroy a window.
z" SDL_DestroyWindow"       1 SDL2.dll DestroyWindow ( window -- )
```

## SDL_Color -- n

Définit l'espace nécessaire pour une structure SDL_Color.

```
create border-color SDL_Color allot
  $ff $00 $00 border-color SDL_Color!
```

## SDL_Color! r g b addr --

Affecte des valeurs de couleur r g b dans une structure `SDL_Color`.

```
create border-color SDL_Color allot
  $ff $00 $00 border-color SDL_Color!
```

## SDL_INIT_VIDEO -- n

Constante. Indique à la SDL que vous voulez initialiser le sous-système vidéo.

```
SDL_INIT_VIDEO SDL.Init
```

## SetRenderDrawColor renderer r g b a -- fl

Définir la couleur utilisée pour les opérations de dessin (Rect, Ligne et Effacer)

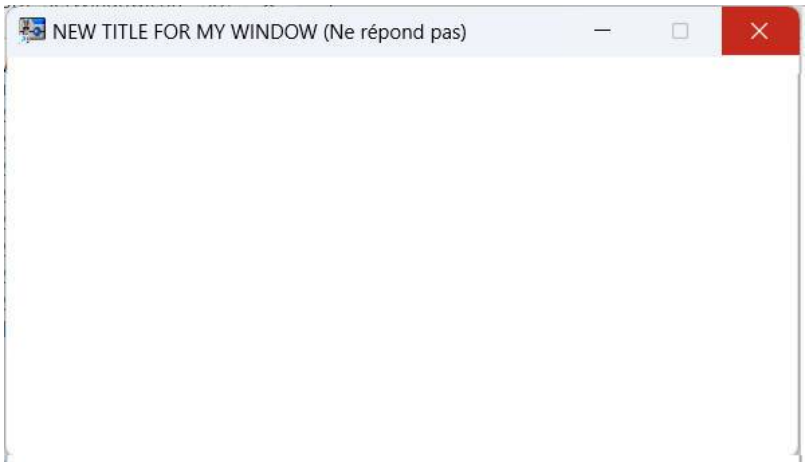
Paramètres

- **renderer** le contexte de rendu
- **r** la valeur rouge utilisée pour dessiner sur la cible de rendu
- **g** la valeur verte utilisée pour dessiner sur la cible de rendu
- **b** la valeur bleue utilisée pour dessiner sur la cible de rendu
- **a** la valeur alpha utilisée pour dessiner sur la cible de rendu ; généralement `SDL_ALPHA_OPAQUE` (255). Utilisez `SetRenderDrawBlendMode` pour spécifier comment le canal alpha est utilisé

## SetWindowIcon window icon

Définit l'icône d'une fenêtre.

```
z" SDL2/logo01.bmp" constant ZFILE
0 value IMAGE01     \ is SDL_Structure ??
ZFILE SDL.load-image to IMAGE01
WIN0 IMAGE01 SetWindowIcon drop
```




## SetWindowSize window w h --

Définit la taille de la zone client d'une fenêtre.

Paramètres :

- **window** la fenêtre à modifier
- **w** la largeur de la fenêtre en pixels, dans les coordonnées de l'écran, doit être > 0/li>
- **h** la hauteur de la fenêtre en pixels, dans les coordonnées de l'écran, doit être > 0/li>

La taille de la fenêtre en coordonnées de l'écran peut différer de la taille en pixels. Utilisez `GetRendererOutputSize` pour obtenir la taille réelle de la zone client en pixels.

```
WIN0 400 200 SetWindowSize drop
```

## SetWindowTitle window zstr -- 0|err

Donne un titre à une fenêtre.

```
WIN0 z" NEW TITLE FOR MY WINDOW" SetWindowTitle drop
REN0 RenderClear drop
REN0 RenderPresent drop
```

## ShowWindow window -- fl

Montre une fenêtre.