

Références SDL2 pour eForth Windows

version 1.1 - dimanche 3 novembre 2024



Auteur

- Marc PETREMANN

Contents

Auteur.....	1
SDL2.....	3
CreateRenderer window index flag -- render.....	3
CreateWindow zstr x y w h fl -- win.....	3
Delay ms -- fl.....	4
DestroyRenderer render -- fl.....	4
DestroyWindow win -- fl.....	4
GetError -- n.....	4
GetTicks -- ms.....	4
GetWindowFlags window -- win-flag.....	4
GetWindowSize windows *w *h -- fl.....	4
GetWindowSizeInPixels windows *w *h -- fl.....	5
HideWindow window -- fl.....	5
Init n -- n.....	5
LoadBMP file -- surface.....	6
LoadBMP_RW RWops freesrc -- 0 surface.....	6
MaximizeWindow window -- fl.....	6
MinimizeWindow windows -- fl.....	6
Quit --.....	6
RaiseWindow window -- fl.....	7
RenderClear render -- 0 err.....	7
RenderDrawLine render x0 y0 x1 y1 -- 0 err.....	7
RenderDrawLines render *points count -- 0 err.....	7
RenderDrawRect render *rect --.....	8
RenderDrawRects render *rect count.....	8
RenderFillRect render *rect -- 0 err.....	9
RenderGetLogicalSize render *width *height -- 0 err.....	10
RenderPresent render --.....	10
RenderSetLogicalSize render w h -- 0 err.....	10
RestoreWindow window -- fl.....	10
RWFromFile file mode -- 0 RWops.....	10
SDL2.dll -- <name>.....	11
SDL_Color -- n.....	11
SDL_Color! r g b addr --.....	11
SDL_INIT_VIDEO -- n.....	11
SetRenderDrawColor renderer r g b a -- fl.....	11
SetWindowSize window w h --.....	12
ShowWindow window -- fl.....	12

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 !
```

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**

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

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 initialize! 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 previously initialized with CreateRenderer
REN0 RenderClear drop
```

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

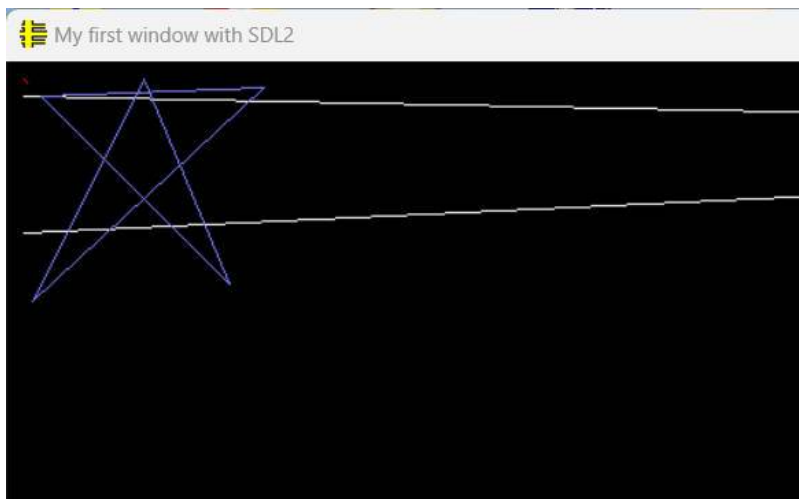
Renvoie 0 en cas de réussite ou un code d'erreur négatif en cas d'échec.

```
\ 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,

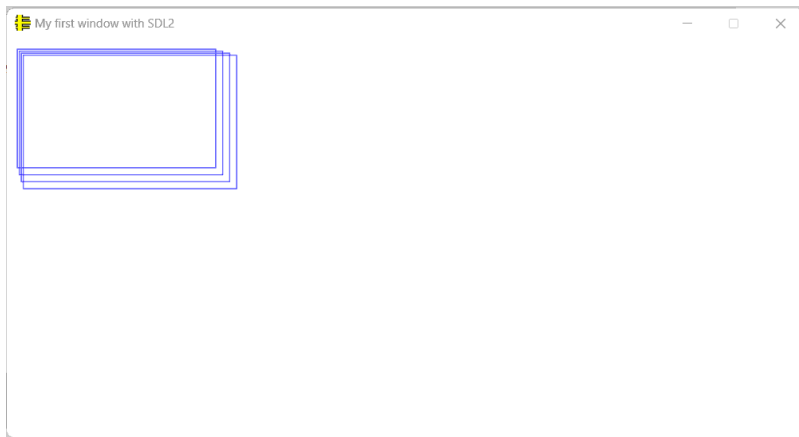
REN0 63 63 255 255 SetRenderDrawColor drop
REN0 RECTS RenderDrawRect drop
REN0 RenderPresent drop
```

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.

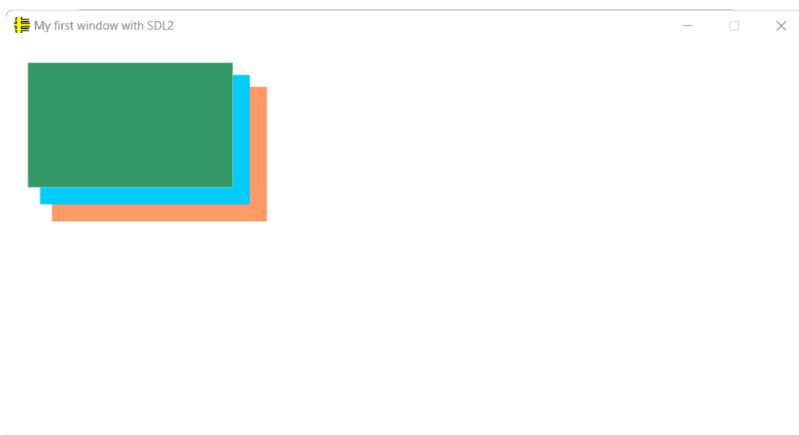
```
create RECTS
    10 L,   10 L,  200 L,  120 L,
    22 L,   22 L,  205 L,  125 L,
    34 L,   34 L,  210 L,  130 L,
    46 L,   46 L,  215 L,  135 L,

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 --`

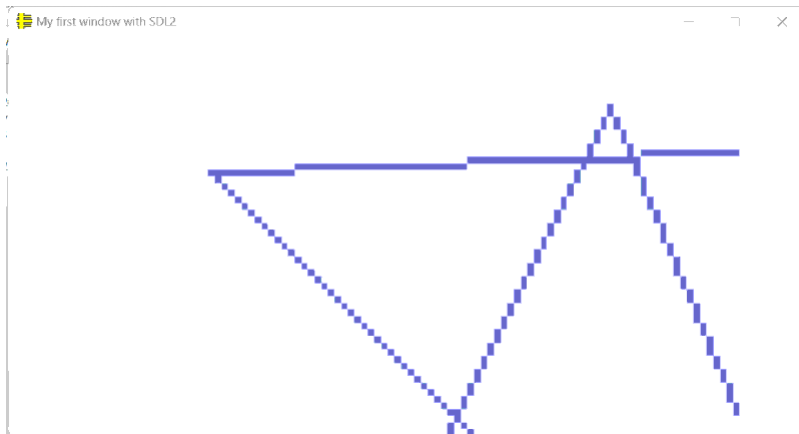
Mettre à jour l'écran avec tout rendu effectué depuis l'appel précédent.

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 rendererer r g b a -- fl

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

Paramètres

- **rendererer** 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

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
- **h** la hauteur de la fenêtre en pixels, dans les coordonnées de l'écran, doit être > 0

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
```

ShowWindow **window -- fl**

Montre une fenêtre.