Liste fonction SDL2

- int SDL_Init(Uint32 flags) : charge les modules nécessaires pour lancer SDL, fonction critique

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_EVERYTHING	all of the above subsystems
SDL_INIT_NOPARACHUTE	compatibility; this flag is ignored

- void SDL_Quit(void) : libère les ressources utilisées par SDL
- char* SDL_GetError(void) : retourne la dernière erreur
- SDL_Window* SDL_CreateWindow(const char* title, int x, int y, int w, int h, Uint32 flags) : créer une fenêtre à la position x y, de taille w h. , fonction critique

SDL_WINDOW_FULLSCREEN	fullscreen window
SDL_WINDOW_FULLSCREEN_DESKTOP	fullscreen window at the current desktop resolution
SDL_WINDOW_OPENGL	window usable with OpenGL context
SDL_WINDOW_SHOWN	window is visible
SDL_WINDOW_HIDDEN	window is not visible
SDL_WINDOW_BORDERLESS	no window decoration
SDL_WINDOW_RESIZABLE	window can be resized
SDL_WINDOW_MINIMIZED	window is minimized
SDL_WINDOW_MAXIMIZED	window is maximized
SDL_WINDOW_INPUT_GRABBED	window has grabbed input focus
SDL_WINDOW_INPUT_FOCUS	window has input focus
SDL_WINDOW_MOUSE_FOCUS	window has mouse focus
SDL_WINDOW_FOREIGN	window not created by SDL
SDL_WINDOW_ALLOW_HIGHDPI	window should be created in high-DPI mode if supported (>= SDL 2.0.1)
SDL_WINDOW_MOUSE_CAPTURE	window has mouse captured (unrelated to INPUT_GRABBED, >= SDL 2.0.4)
SDL_WINDOW_ALWAYS_ON_TOP	window should always be above others (X11 only, >= SDL 2.0.5)
SDL_WINDOW_SKIP_TASKBAR	window should not be added to the taskbar (X11 only, >= SDL 2.0.5)
SDL_WINDOW_UTILITY	window should be treated as a utility window (X11 only, >= SDL 2.0.5)
SDL_WINDOW_TOOLTIP	window should be treated as a tooltip (X11 only, >= SDL 2.0.5)
SDL_WINDOW_POPUP_MENU	window should be treated as a popup menu (X11 only, >= SDL 2.0.5)

- void SDL_DestroyWindow(SDL_Window* window) : libère la mémoire de la fenêtre.
- void SDL_SetWindowIcon(SDL_Window* window, SDL_Surface* icon) : rajoute une icône sur la fenêtre
- SDL_Renderer* SDL_CreateRenderer(SDL_Window* window, int index, Uint32 flags) : créer un renderer, fonction critique
- void SDL_DestroyRenderer(SDL_Renderer* renderer) : détruit un renderer
- int SDL_CreateWindowAndRenderer(int width, int height, Uint32 window_flags, SDL_Window** window, SDL_Renderer** renderer) : créer une fenêtre et un renderer en même temps, fonction critique
- char* SDL_GetWindowTitle(SDL_Window* window) : récupère le titre d'une fenêtre.
- void SDL_Delay(Uint32 ms) : met en place une pause dans le programme
- int SDL_SetRenderDrawColor(SDL_Renderer* renderer, int r, int g, int b, int a) : choisi une couleur à utiliser pour le renderer
- int SDL_RenderClear(SDL_Renderer* renderer) : nettoie le contenu actuel de la fenêtre
- void SDL_RenderPresent(SDL_Renderer* renderer) : met à jour le contenu de la fenêtre
- int SDL_RenderDrawPoint(SDL_Renderer* renderer, int x, int y) : dessine un point aux coordonnées x y

- int SDL_RenderDrawPoints(SDL_Renderer* renderer, const SDL_Point* points, int count) : dessine plusieurs points
- int SDL_RenderDrawRect(SDL_Renderer* renderer, const SDL_Rect* rect) : dessine le contour d'un rectangle
- int SDL_RenderFillRect(SDL_Renderer* renderer, const SDL_Rect* rect) : dessine un rectangle remplie
- typedef struct SDL_Rect
  {
  int x, y;
  int w, h;
  };
- int SDL_RenderDrawLine(SDL_Renderer* renderer, int x1, int y1, int x2, int y2) : trace une ligne
- int SDL_SetRenderDrawBlendMode(SDL_Renderer* renderer, SDL_BlendMode blendMode) : gère la transparence d'un dessin
- SDL_bool SDL_PointInRect(const SDL_Point* p, const SDL_Rect* r) : vérifie si un point est dans un rectangle
- SDL_bool SDL_HasIntersection(const SDL_Rect* A, const SDL_Rect* B) : vérifie si 2 rectangles ont une intersection
- SDL_bool SDL_IntersectRect(const SDL_Rect* A, const SDL_Rect* B, SDL_Rect* result) : vérifie si 2 rectangles ont une intersection et donne le résultat de cette intersection si elle existe.
- SDL_bool SDL_IntersectRectAndLine(const SDL_Rect* rect, int *x1, int *y1, inbt *x2, int *y2) : intersection entre une ligne et un rectangle
- SDL_Texture* SDL_CreateTexture(SDL_Renderer* renderer, Uint32 format, int access, int w, int h) : créer une texture de la taille donnée, liste format (https://wiki.libsdl.org/SDL2/SDL_PixelFormatEnum) , liste access dans le tableau.

SDL_TEXTUREACCESS_STATIC	changes rarely, not lockable
SDL_TEXTUREACCESS_STREAMING	changes frequently, lockable
SDL_TEXTUREACCESS_TARGET	can be used as a render target

- void SDL_DestroyTexture(SDL_Texture* texture) : détruit une texture
- int SDL_SetRenderTarget(SDL_Renderer* renderer, SDL_Texture* texture) : demande au rendu de dessiner la texture
- int SDL_RenderCopy(SDL_Renderer* renderer, SDL_Texture* texture, const SDL_Rect* srcrect, const SDL_Rect* dstrect) : affiche la texture (en prenant dessus srcrect) sur la fenêtre à la place dstrect.
- SDL_Surface* SDL_CreateRGBSurface(Uint32 flags, int width, int height, int depth, Uint32 Rmask, Uint32 Gmask, Uint32 Bmask, Uint32 Amask) : créer une surface
- void SDL_FreeSurface(SDL_Surface* surface) : libère la mémoire propre à la surface
- int SDL_FillRect(SDL_Surface* dst, const SDL_Rect* rect, Uint32 color) : remplie un rectangle dans la surface d'une couleur donnée
- Uint32 SDL_MapRGBA(const SDL_PixelFormat* format, Uint8 r, Uint8 g, Uint8 b, Uint8 a) : retourne une couleur sous le format Uint32
- SDL_Surface* SDL_LoadBMP(const char* file) : charge une image depuis un fichier (.bmp), fonction critique
- SDL_Texture* SDL_CreateTextureFromSurface(SDL_Renderer* renderer, SDL_Surface* surface) : créer une texture depuis une surface
- int SDL_QueryTexture(SDL_Texture* texture, Uint32* format, int* access, int* w, int* h) : récupère la taille de la texture dans w et h.
- SDL_Surface * IMG_Load(const char *file) : charge une image de n'importe quel format, fonction critique
- int SDL_WaitEvent(SDL_Event* event) : gère les évènements bloquant (liste event : https://wiki.libsdl.org/SDL2/SDL_Event)
- int SDL_PollEvent(SDL_Event* event) : gère les évènements non bloquant
- Evènement fenêtre : https://wiki.libsdl.org/SDL2/SDL_WindowEventID
- Evènement clavier : https://wiki.libsdl.org/SDL2/SDL_Keycode
- Evènement souris : https://wiki.libsdl.org/SDL2/SDL_MouseMotionEvent , https://wiki.libsdl.org/SDL2/SDL_MouseButtonEvent , https://wiki.libsdl.org/SDL2/SDL_MouseWheelEvent

- int Mix_OpenAudio(int frequency, Uint16 format, int channels, int chunksize) : initialise la librairie audio, fonction critique
- void Mix_CloseAudio(void) : libère la mémoire pour la librairie audio
- Mix_Music * Mix_LoadMUS(const char *file) : charge une musique, fonction critique
- int Mix_PlayMusic(Mix_Music *music, int loops) : joue la musique chargé précédemment
- void Mix_PauseMusic(void) : met en pause la musique
- void Mix_ResumeMusic(void) : relance la musique
- void Mix_RewindMusic(void): remet la musique au début
- int Mix_VolumeMusic(int volume) : change le volume de la musique
- char* Mix_GetError() : erreur de la lib musique
- int TTF_Init(void) : charge la lib texte, fonction critique
- void TTF_Quit(void) : libère la mémoire de la lib texte
- TTF_Font* TTF_OpenFont(const char *file, int ptsize) : charge une police, fonction critique
- void SDLCALL TTF_CloseFont(TTF_Font *font) : libère la mémoire pour une police
- SDL_Surface * TTF_RenderText_Solid(TTF_Font *font, const char *text, SDL_Color fg) : écrit du texte sans antiliasing
- SDL_Surface * TTF_RenderText_Shaded(TTF_Font *font, const char *text, SDL_Color fg, SDL_Color bg) : écrit du texte sans transparence pour le fond et avec antialiasing.
- SDL_Surface * TTF_RenderText_Blended(TTF_Font *font, const char *text, SDL_Color fg) : écrit du texte avec antialiasing