Partie 1

Question 1 : Qu’est-ce que la SDL 2 ?

La SDL 2 (ou Simple DirectMedia Layer) est une bibliothèque logicielle permettant de gérer l’affichage vidéo et l’audio.

Question 2 : Que peut faire la SDL 2 ?

La SDL 2 permet d’afficher un rendu graphique 2D, utile à la création de jeux vidéo ou d’interfaces graphiques.

Question 3 : Sur quelles plateformes fonctionne la SDL 2 ?

La SDL 2 fonctionne officiellement sur Windows, Mac OS X, Linux, iOS, et Android.

Question 4 : Comment obtenir et installer la SDL 2 sur un EDI ? Donnez les étapes de la configuration d’un projet SDL2 avec CodeBlocks (gcc).

Tout d’abord, nous devons récupérer l’archive contenant SDL 2 sur le siteUne image contenant capture d’écran

Description générée automatiquement

A partir d’ici, nous téléchargerons l’archive SDL2-devel-2.0.10-mingw.tar.gz. On extrait le fichier, que l’on place ensuite dans un dossier prévu pour accueillir cette librairie (par exemple dans un dossier SDL\_2 dans programmes files, à la racine du fichier CodeBlocks)

Puis nous ouvrons CodeBlocks, dans un nouveau projet, on clique sur ‘Settings’ puis ‘Compiler’, on sélectionne ‘Linker Settings’ et on y ajoute les chemins de notre librairie comme tel :

* C:\Program Files (x86)\CodeBlocks\SDL2-2.0.10\i686-w64-mingw32\lib\libSDL2.a
* C:\Program Files (x86)\CodeBlocks\SDL2-2.0.10\i686-w64-mingw32\lib\libSDL2.dll.a

Attention à bien sélectionner le fichier correspondant à votre ordinateur : i686 pour les processeurs 32 bit, ou x86-64 pour les processeurs 64 bits.

Un fois ceci fait, on appuie sur ok et on va sur l’onglet ‘Search directories’. Dans l’onglet ‘Compiler’ on entre les chemins pour indiquer au compilateur où se situe le dossier include et le dossier lib pour la SDL2.

* C:\Program Files (x86)\CodeBlocks\SDL2-2.0.10\i686-w64-mingw32\include
* C:\Program Files (x86)\CodeBlocks\SDL2-2.0.10\i686-w64-mingw32\lib

Question 5 : Quelles sont les différences entre SDL 1 et SDL 2 ?

SDL 1 utilise le processeur, alors que SDL 2 utilise les ressources carte graphique. Cette dernière permet de gérer plusieurs fenêtres, de gérer leurs tailles et les boites de dialogues.

Question 6 : Pour utiliser la SDL 2 dans votre projet, vous devez initialiser l’instance SDL 2, quelles sont les étapes d’initialisation à réaliser pour utiliser la lib SDL 2, donnez le code correspondant et expliquez précisément chaque instruction.

* Voir code

Question 7 : Comment créer une fenêtre avec SDL 2, donnez et expliquez le code correspondant à la création d’une fenêtre avec SDL 2.

* Voir code

Question 8 : Gestion d’une fenêtre SDL 2, donnez les primitives associées à la fenêtre SDL 2.

SDL\_Window\* SDL\_CreateWindow(const char\* title, int x, int y, int w, int h, Uint32 flags) ;

* Title : il s’agit du titre (chaîne de caractère) de la fenêtre créer par SDL2 ;
* X : la position en X de la fenêtre SDL2 sur l’écran ;
* Y : la position en Y de la fenêtre SDL2 sur l’écran ;
* W : la largeur W de la fenêtre SDL2 ;
* H : la hauteur H de la fenêtre SDL2 ;
* Flags : l’option d’affichage de la fenêtre SDL2 (SHOWN, BORDERLESS, etc.) ;

Question 9 : Gérer un rendu : qu’est-ce qu’un renderer, comment est-il associé à la fenêtre SDL 2 ? Donnez et expliquez le code permettant de gérer un rendu, création, affichage, effacement….

* Voir code

Question 10 : SDL\_Point et SDL\_Rect : Donnez et expliquez le code de la déclaration d’un SDL\_Point et SDL\_Rect.

* Voir code

Question 11 : Les couleurs avec la SDL2 : donnez les instructions permettant de gérer la couleur avec la SDL 2.

SDL\_SetRenderDrawColor(SDL\_Renderer\* renderer, Uint8 r, Uint8 g, Uint8 b, Uint8 a) ;

* Renderer : le renderer à sélectionner pour afficher la couleur ;
* R : la valeur pour la couleur rouge, allant de 0 à 255 ;
* G : la valeur pour la couleur verte, allant de 0 à 255 ;
* B : la valeur pour la couleur bleue, allant de 0 à 255 ;
* A : la valeur pour la transparence, allant de 0 à 255. On peut aussi remplacer le 255 par SDL\_ALPHA\_OPAQUE ;

On doit ensuite afficher le renderer avec SDL\_RenderPresent(pRenderer).

Question 12 : Donnez le code permettant d’afficher un fond rouge dans le rendu.

* Voir code

Question 13 : Dessiner dans le rendu : donnez le code des fonctions permettant de dessiner dans le rendu les formes suivantes : Carré vide - Carré plein - Cercle vide - Cercle plein.

* Voir code

Question 14 : Quelles sont les fonctions permettant de dessiner des points et des lignes ?

SDL\_RenderDrawPoint(SDL\_Renderer\* renderer, int x, int y) ;

* Renderer : le renderer à sélectionner pour afficher le point ;
* X : la coordonnée X du point sur la fenêtre ;
* Y : la coordonnée Y du point sur la fenêtre ;

et

SDL\_RenderDrawLine(SDL\_Renderer\* renderer, int x1, int y1, int x2, int y2) ;

* Renderer : le renderer à sélectionner pour afficher la couleur ;
* X1 : la coordonnée X1 du premier point sur la fenêtre ;
* Y1 : la coordonnée Y1 du premier point sur la fenêtre ;
* X2 : la coordonnée X2 du second point sur la fenêtre ;
* Y2 : la coordonnée Y2 du second point sur la fenêtre ;

Question 15 : A quoi servent les fonctions SDL\_RenderClear et SDL\_RenderPresent ?

SDL\_RenderClear : efface la cible du rendu actuelle ;

SDL\_RenderPresent : met à jour l’écran avec le rendu ;

Question 16 : Expliquez SDL\_Delay.

SDL\_Delay permet d’attendre un certain nombre de milliseconde avant de continuer le programme.

Question 17 : Qu’est-ce qu’une surface en SDL 2.

Les surfaces sont l’équivalent des textures, mais bien moins performantes que ces dernières. Cependant, les surfaces peuvent être modifiées pixel par pixel plus facilement que sur les textures.

Question 18 : Donnez le code permettant de créer une surface.

SDL\_Surface \*surface = NULL ;

surface = SDL\_CreateRGBSurface(0, 300, 200, 32, 0, 0, 0, 0) ;

SDL\_FreeSurface(surface) ;

Question 19 : Dessiner dans une surface : Donnez le code de la fonction SDL\_FillRect.

SDL\_FillRect(SDL\_Surface\* dst, const SDL\_Rect\* rect, Uint32 color) ;

* Dst : la surface à sélectionner pour afficher le remplissage ;
* Rect : la portion de la surface à remplir (NULL si toute la surface doit être remplie) ;
* Color : la couleur sélectionnée avec SDL\_MapRGBA() ;

Question 20 : SDL\_BlitSurface(..) : donnez le code pour tester cette méthode.

SDL\_BlitSurface(SDL\_Surface\* src, const SDL\_Rect\* srcrect, SDL\_Surface\* dst, SDL\_Rect\* dstrect) ;

* Src : la surface à copier ;
* Srcrect : la portion du rectangle à copier (NULL si toute la surface doit être copiée) ;
* Dst : la surface cible sur laquelle on va copier le rectangle ;
* Dstrect : la portion de la surface à remplir ;

Question 21 : Les textures : Qu’est-ce qu’une texture ? Donnez le code permettant de créer une texture.

SDL\_Texture\* SDL\_CreateTexture(SDL\_Renderer\* renderer, Uint32 format, int access, int w, int h)

* Renderer : le renderer à sélectionner pour afficher la texture ;
* Format : voir SDL\_PixelFormatEnum ;
* Access : voir SDL\_TextureAccess ;
* W : la largeur W de la texture ;
* H : la hauteur H de la texture ;

Question 22 : Dessinez dans une texture : SDL\_RenderTarget, donnez le code pour tester cette méthode.

SDL\_SetRenderTarget(SDL\_Renderer\* renderer, SDL\_Texture\* texture) ;

* Renderer : le renderer à sélectionner pour afficher la texture ;
* Texture : la texture à afficher ;

Question 23 : SDL\_RenderCopy : A quoi sert cette méthode, donnez le code pour tester cette méthode.

SDL\_RenderCopy est très similaire à SDL\_BlitSurface. Il utilise le renderer à la place de la surface, et la texture au lieu du rectangle.

* SDL\_RenderCopy(SDL\_Renderer\* renderer, SDL\_Texture\* texture, const SDL\_Rect\* srcrect, const SDL\_Rect\* dstrect) ;

Question 24 : SDL\_QueryTexture : A quoi sert cette méthode, donnez le code pour tester cette méthode.

SDL\_QueryTexture(SDL\_Texture\* texture, Uint32\* format, int\* access, int\* w, int\* h) ;

Question 25 : SDL\_LoadBMP, est la méthode permettant d’utiliser des bmp, donnez le code permettant de tester cette fonction.

* Voir code