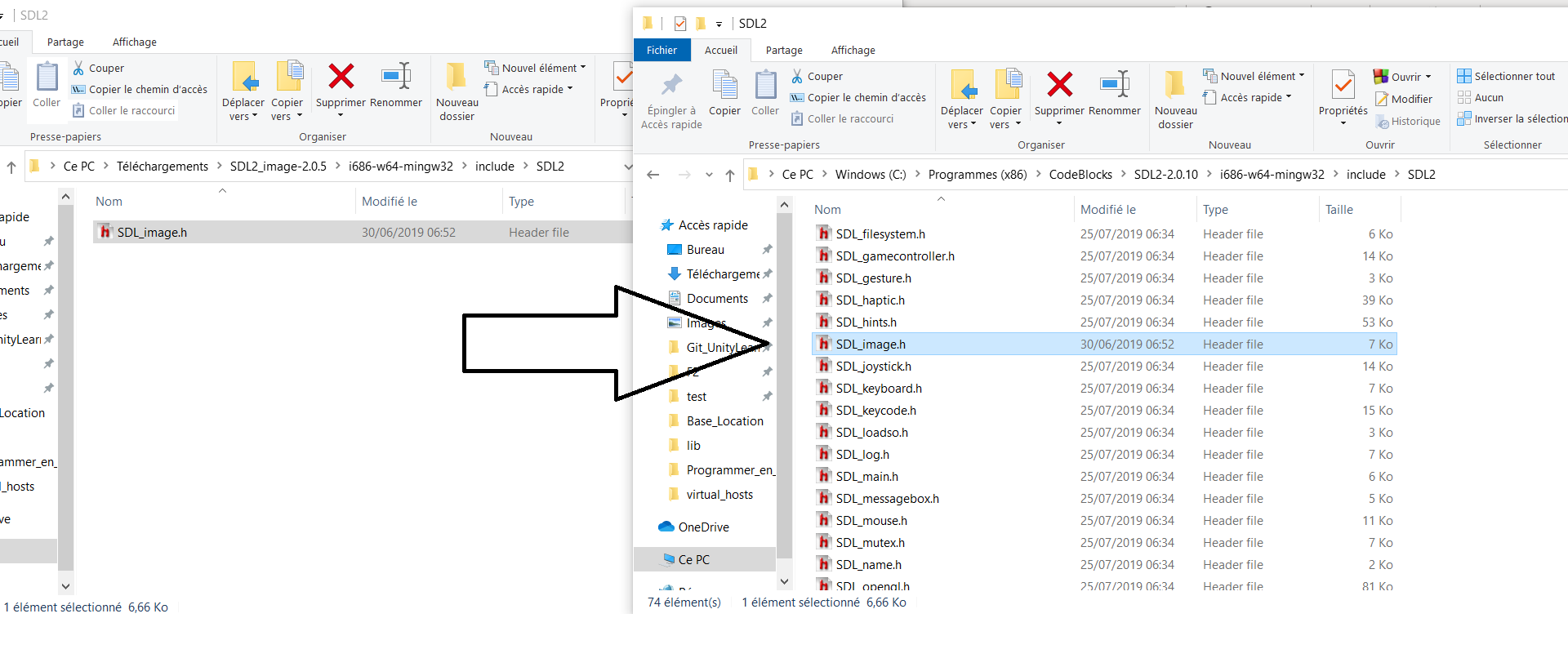
**1) Quel Api faut-il ajouter à votre projet de développement afin de pouvoir prendre en charge les images au format (JPG, PNG….), que fait cet API? 10 points**

L’API à ajouter au projet s’appelle SDL\_Image : elle charge les images en tant que surfaces et textures. Les format supportés sont les suivants : BMP, BIG, JPEG, LBM, PCX, PNG, PNM, SVG, TGA, TIFF, WEBP, XCF, XPM, XV.

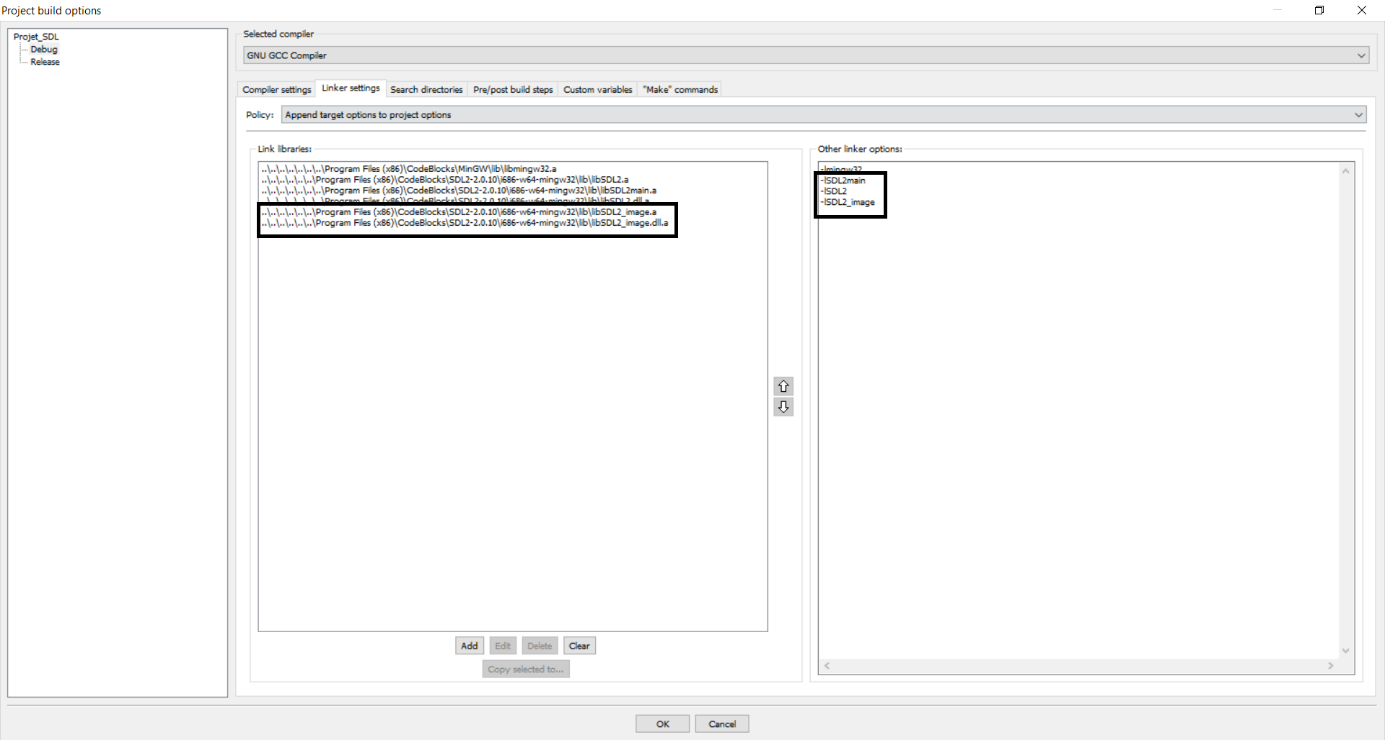
**2) Donnez les étapes de configuration pour l’utilisation de ce nouvel API. 10 points**

- Se rendre sur le site : <https://www.libsdl.org/projects/SDL_image/>  
- Télécharger « SDL2\_image-devel-2.0.5-mingw.tar.gz (MinGW 32/64-bit) » (Development Libraries)  
- Copier le contenu de chaque dossier contenu dans « i686-w64-mingw32 » dans le dossier du même nom contenu au sein du dossier SDL-2.0.10   
- Copier toutes les .dll contenues dans le dossier « bin » directement à la racine du projet



*Exemple avec le fichier SDL\_image.h*

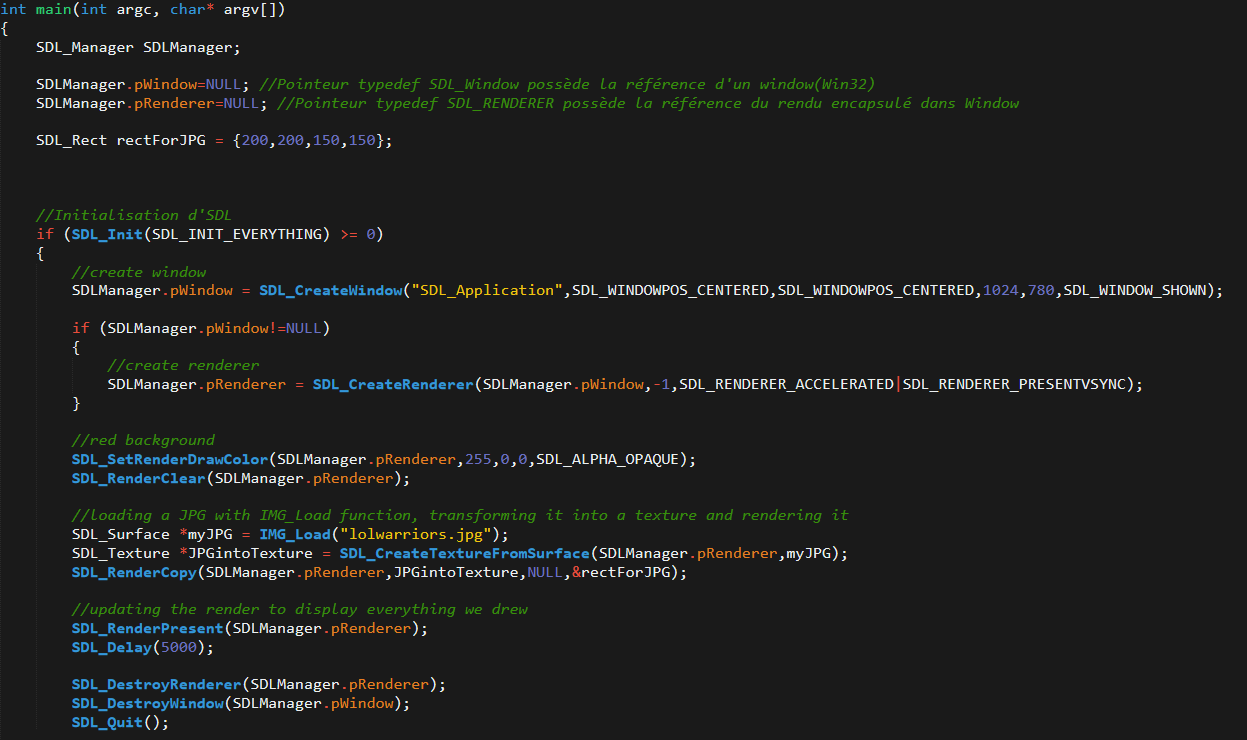
- Dans CodeBlocks, aller dans « Project => Build Options », cliquer sur « Debug » et aller dans « Linker settings »

  
Rajouter les deux blocs encadrés en noir

Et ne pas oublier « #include <SDL2/SDL\_Image.h> » dans le projet !

3) **Donnez un exemple de code en c chargeant une image au format png dans la fenêtre (image de votre choix). Commentez et expliquez ce code. 10 points**

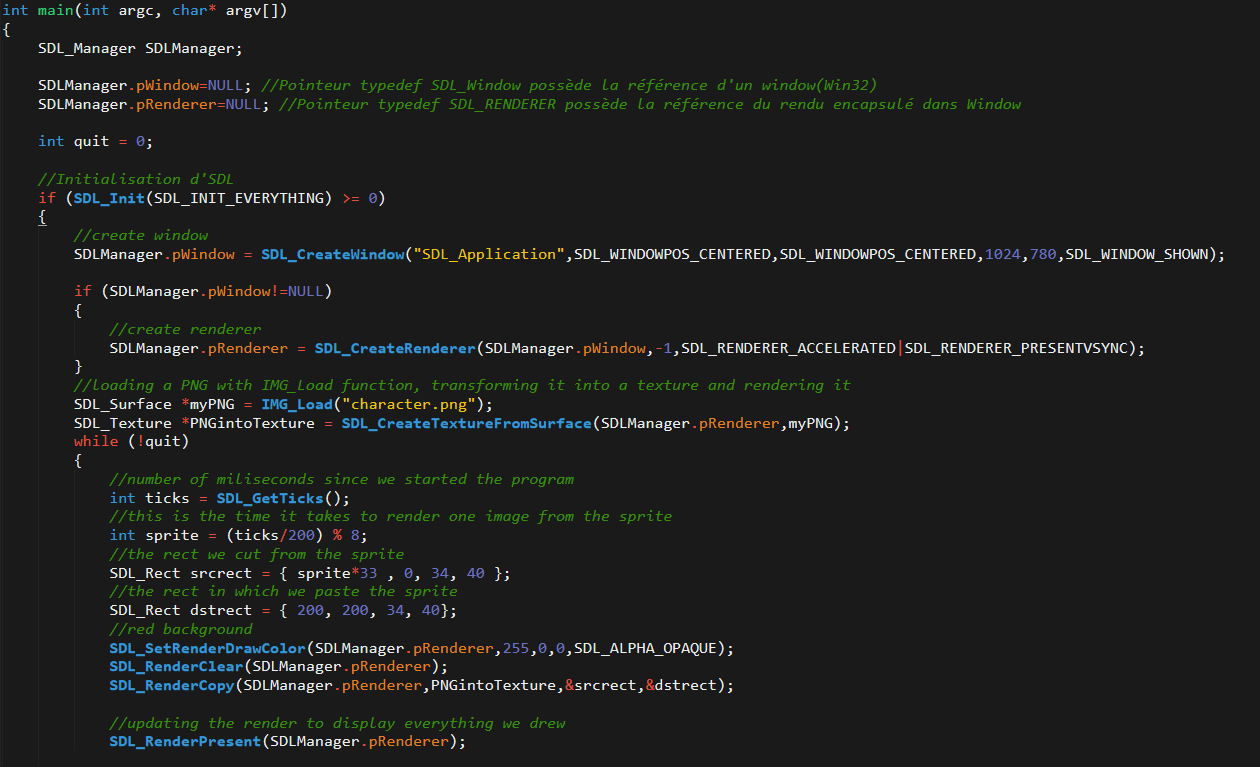
J’ai fais pour une image JPG mais c’est le même principe.



Ca marche pareil que lors de la création d’une surface : on utilise la fonction IMG\_Load en passant en paramètre le chemin vers l’image à charger. Elle est chargée en tant que surface : on la transforme en texture et on utilise SDL\_RenderCopy pour la passer dans le render, et on l’affiche en appelant SDL\_RenderPresent().

**4) Les images animées : Gestion des Tileset Expliquez ce qu’est un tileset, à quoi cela peut servir? 10 points.**

Un tileset est une collection d’images rectangulaires (ou bien carrées) affichées à l’écran. Le tileset correspond à l’entièreté des images utilisées pour créer le rendu.

5) **Créez une image animée dans votre fenêtre.(Vous simulez les changements d’animation à chaque frame). 15 points**

**6) Le Tile Mapping Expliquez ce qu’est le tile mapping, technique de l’image fixe ou tile mapping? Comment contrôler les ressources lorsque l’on manipule des images dans le cadre de la construction des maps de jeux. 10 points**

La technique de l’image fixe consiste à séparer le joueur du décor, le décor étant une image générée (en png par exemple) et collée au background. Ce n’est pas une méthode efficace, étant donné qu’il est très difficile dans un platformer par exemple, de gérer le fait que notre personnage soit sur une plateforme, ou bien dans les airs.. il est difficile d’accéder aux éléments du jeu.

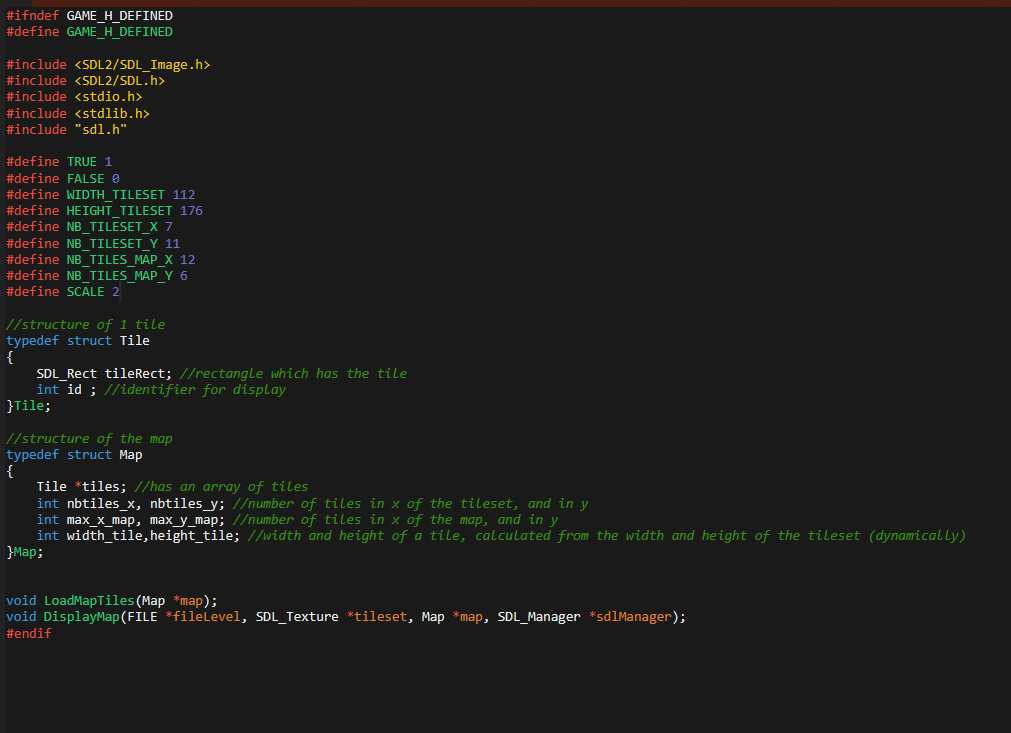
Le tile mapping consiste à coller bout à bout des tiles, le tout en divisant l’écran dans une grille, afin d’avoir un tile par case de grille. De ce fait, chaque élément est accessible étant donné qu’il est généré dynamiquement.

**7) Existe-t-il des logiciels dédiés à la gestion des tiles, si oui nommez-en et expliquez leur fonctionnement. 10 points**

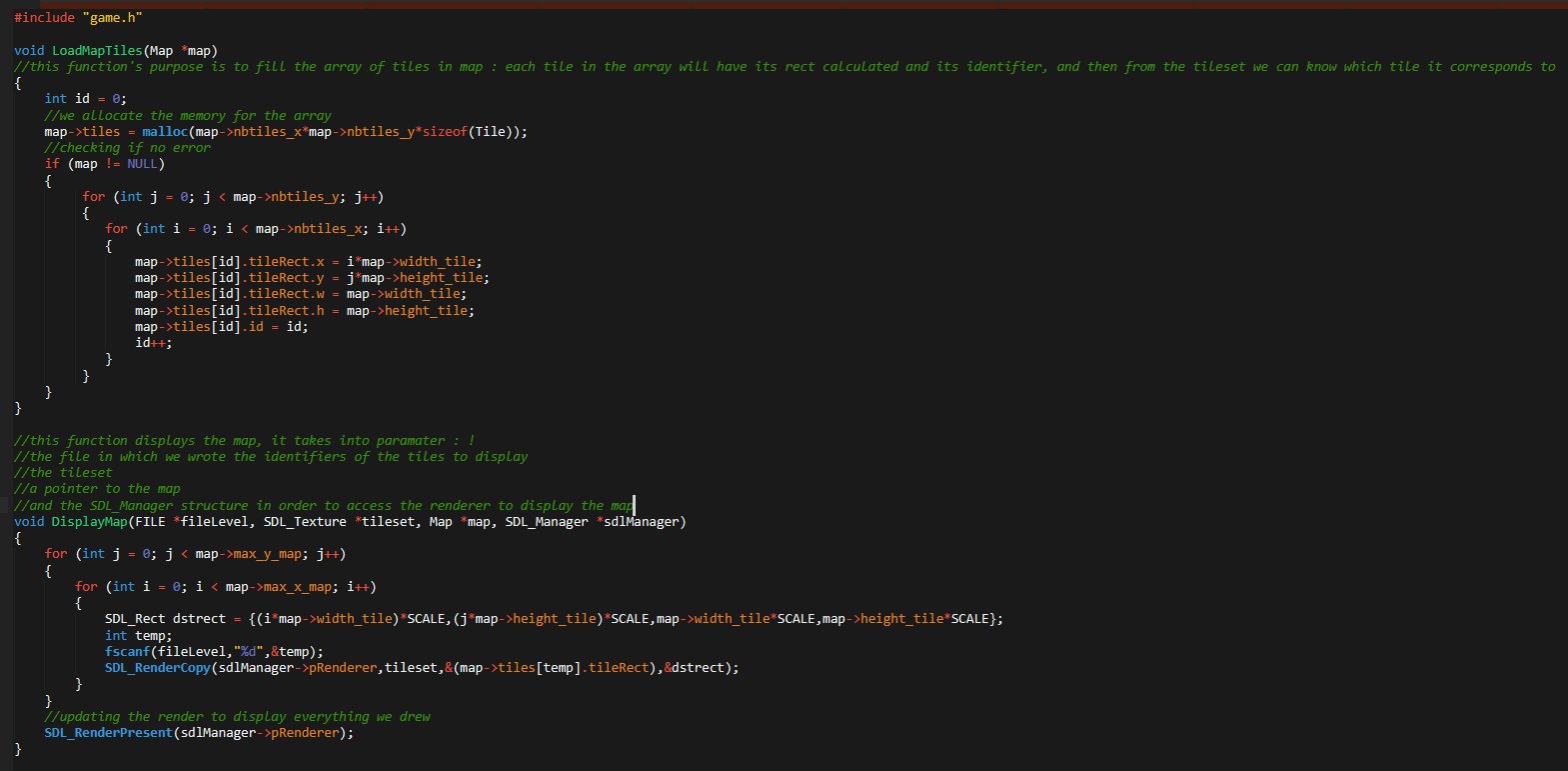
Il existe le logiciel « Tiled » ou bien encore « Tile Studio » : ils nous permettent de charger une image, un tileset, définir les dimensions de chaque tile et ainsi le logiciel découpe le tileset en tiles individuelles que l’on peut utiliser pour créer notre map.

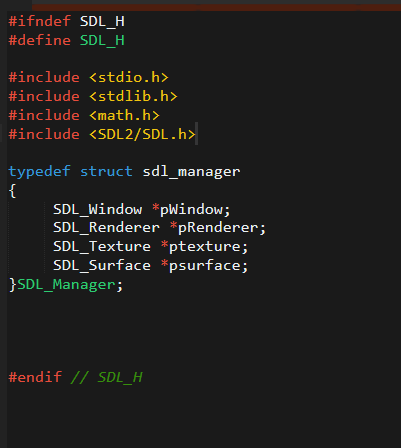
**8) Proposez le code permettant de construire un niveau à partir d’un tileset. (30 points)**

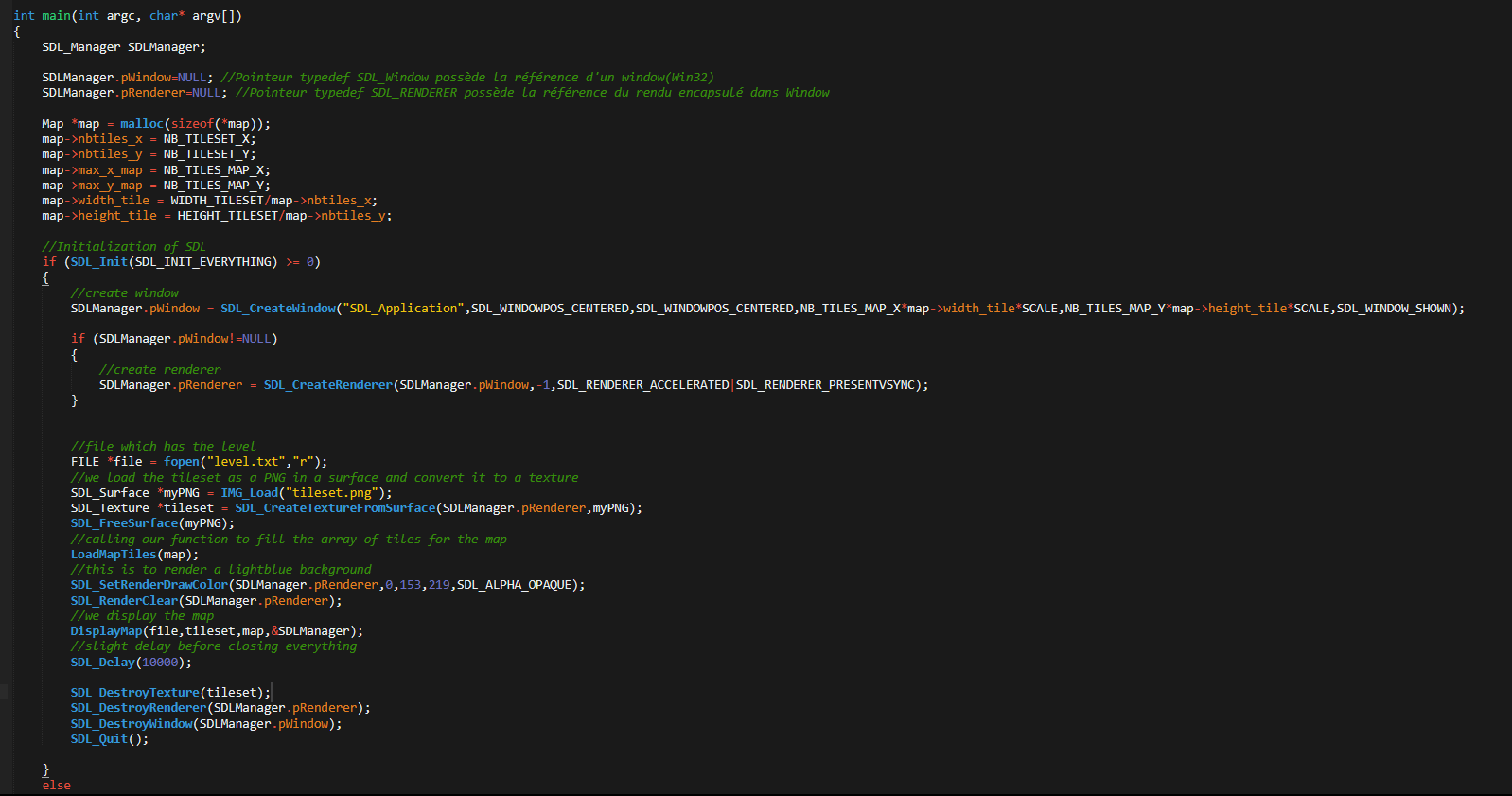
Fichier « game.h »



Fichier « game.c »



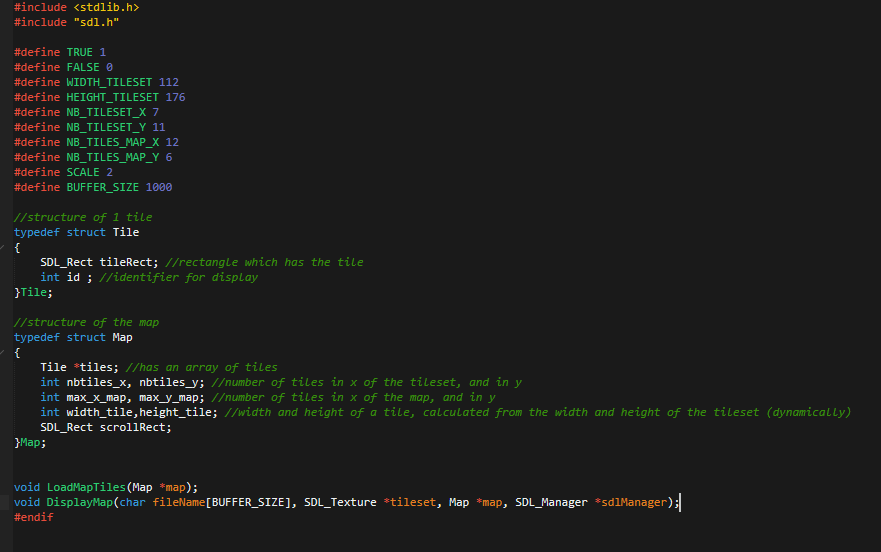
Fichier « sdl.h »  
  
Fichier « main.c »



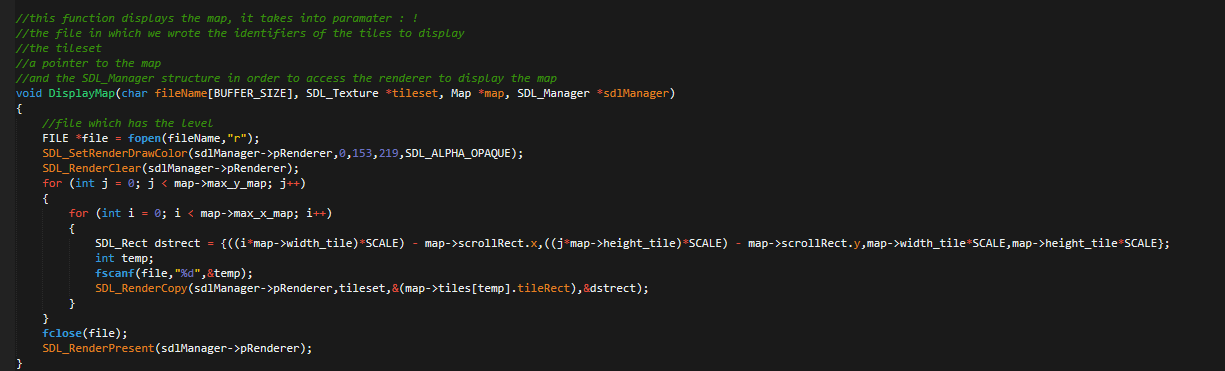
**9) Le Scrolling (Tile Mapping defilement), Expliquez ce qu’est le scrolling dans la création de la map. 10 points**

Lorsqu’on a une très grande carte, on ne veut afficher au joueur qu’une certaine partie, et il doit avancer afin de découvrir le reste. On ne va afficher qu’une partie de la carte, et progressivement afficher le reste quand le joueur avance. C’est ce qu’on appelle le scrolling.

**10) Donnez un exemple de code gérant le scrolling dans votre map. 20 points**Mise à jour du fichier « game.h » : rajout d’un SDL\_Rect pour gérer le scrolling

****

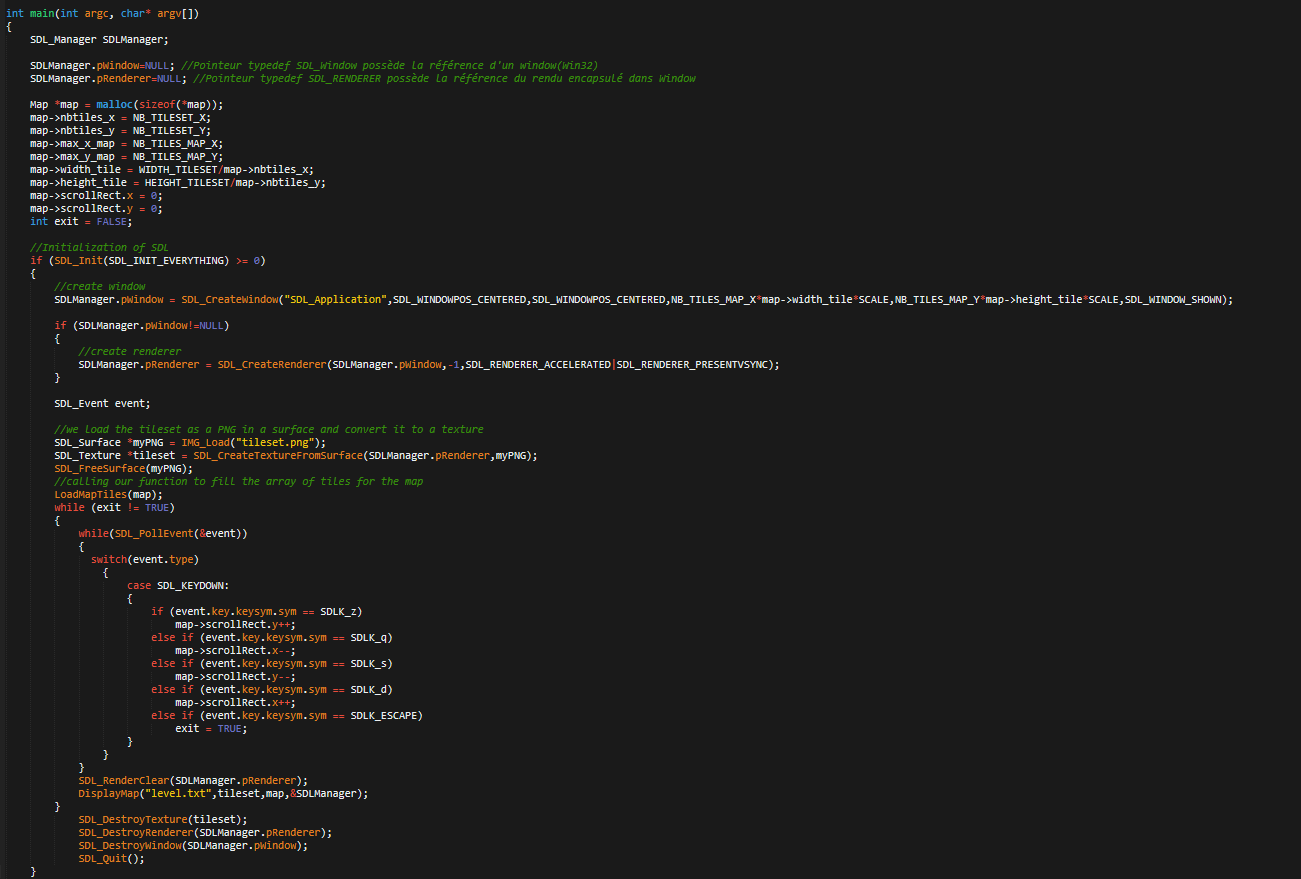
Mise à jour du fichier « game.c » et principalement la fonction d’affichage de la map : on va soustraire, au rectangle où on veut placer la tile, la position x du rectangle de scrolling : ainsi à chaque fois qu’on avance vers la droite (qu’on incrémente la position x de rectScroll) on fais reculer les tiles vers l’arrière et on a ainsi cette sensation de déplacement). J’ai également gérer l’ouverture et la fermeture du fichier à l’intérieur de cette fonction afin d’éviter des problèmes d’affichage suite à la ré-utilisation du fichier.



Mise à jour du fichier « main.c »

Rajout de la boucle principale pour qu’on affiche la carte tant qu’on n’appuie pas sur échap.

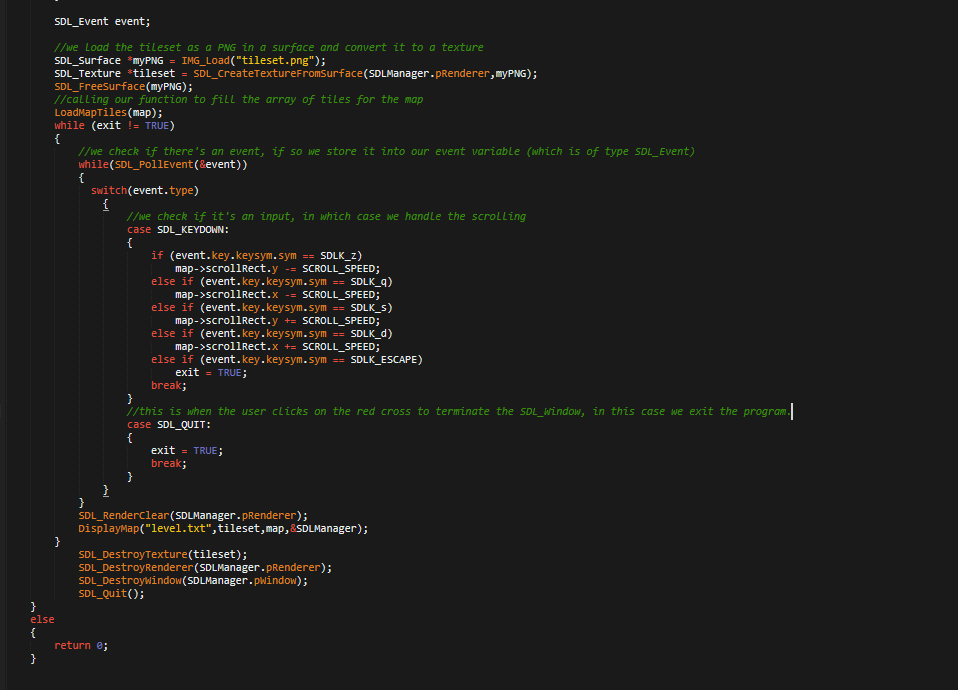
On peut utiliser les touches Z,Q,S,D afin de scroller la map vers le haut, la gauche, le bas et la droite respectivement.



**11)Gestion des events en SLD2, comment est gérée la boucle principale du jeu et la gestion des events avec SDL2 (Quit, pause…). Modifiez votre programme afin de gérer ces events. 10 points**

La boucle principale du jeu va être constituée d’un **while()** qui va attendre qu’un int soit égal à 0 par exemple, et cette valeur sera modifiée lorsque l’on fermera la fenêtre ou bien lorsqu’on appuie sur une touche si l’on veut. Concernant la gestion des évènements, ils sont disponibles dans une queue constituée de SDL\_Event. Pour récupérer un évènement on va déclarer une variable de type SDL\_Event, et appeler la fonction SDL\_PollEvent() ou SDL\_WaitEvent(). SDL\_PollEvent n’est pas bloquante donc même s’il n’y a pas d’évènement actuellement, cela va continuer, alors que SDL\_WaitEvent va attendre jusqu’à qu’un évènement soit lu, auquel cas le programme ne continuera pas.  
Concrètement, on va dans la boucle, voir s’il y a un évènement, gérer l’évènement, clear le Render (donc la frame) d’avant, et actualiser avec la nouvelle frame.

Voici une légère modification du fichier « main.c » où j’ai juste rajouter un #define pour la vitesse de scroll, la gestion d’évènements était déjà présente, j’ai juste rajouté le cas où l’utilisateur clique sur la croix pour fermer le programme (géré par SDL\_Quit).



**12)Input Handling, comment sont gérées les entrées des contrôleurs avec SDL2 , quels types de contrôleurs pouvons-nous utiliser avec SDL2. 10 points**

Comme vu précédemment, les évènements sont stockés dans une queue. Les entrées des contrôleurs font partis de ces évènements : ils ont une structure leur correspondant, qui est passé à notre variable « event » si elle est renvoyée par la fonction SDL\_PollEvent() ou SDL\_WaitEvent(). Avec cette structure on a accès aux différents paramètres de l’input.  
Par exemple avec le code ci-dessus, on regarde si « event.type == SDL\_KEYDOWN », donc entre autre on regarde si l’évènement est de type « appui sur une touche » auquel cas on regarde quel touche va être appuyée et on gère le code en conséquence.   
  
Ainsi, SDL2 nous donne la capacité de gérer : clavier, souris (molette comprise), joystick, input de manettes, et le tactile.

**13)Implémenter la gestion des inputs clavier pour les déplacement haut, bas, gauche et droite dans votre projet. Modifiez votre programme afin de gérer ces events. 15 points**

Etant donné que je gère déjà les inputs clavier pour gérer le scrolling (qui est un déplacement vers les différents axes du rectangle d’affichage), veuillez-vous référer aux 3 questions précédentes (le code n’a pas changé !)

**14)Comment récupérer le keypress de la pression simultanée de deux ou plusieurs touches. Modifiez votre programme afin de donner un exemple de gestion de keypress simultanée. 20 points**

Il existe la fonction SDL\_GetKeyboardState() qui renvoie un pointeur vers un tableau contenant le statut actuel de toutes les touches. (1 si elles sont appuyées, 0 si non)  
Ainsi, on commence par récupérer la référence de ce tableau   
Il ne faut pas toucher à ce pointeur, ainsi on le déclare en « const ». Et il est de type Uint8 donc il faut également déclarer notre variable de ce même type !

Ensuite, dans la boucle principale, on doit juste regarder l’indice de ce tableau à l’index SDL\_SCANCODE (correspondant au SCANCODE de la touche à vérifier) pour gérer le multi-touches !

Ci-dessous le fichier « main.c » avec la boucle de gameplay modifiée : on regarde dans notre tableau si Z et Q sont appuyés en même temps, si c’est le cas alors on se déplace en diagonale en haut à gauche, et ainsi de suite…

