Atelier SDL (Simple Direct Media Layer)

A. Objectif :

Le but de cet atelier est :

1. De vous initier à l'utilisation de la bibliothèque SDL en manipulant plusieurs types de composants multimédias : texte, image et son.
2. Créer un menu simple contenant un background, deux boutons et un texte, avec la présence d'un son continu et un son bref.

Figure 1. Menu

2.1. En lançant l'exécution, l'image du background, le texte « Bienvenue », Les deux boutons « Play » et « Quit» s'affichent, avec le chargement d'un son continu.

2.2. En cliquant sur le bouton fermer de la fenêtre principale, on quitte le menu.

2.3. En cliquant sur le bouton « Quit », on quitte le menu.

2.4. En survolant sur le bouton « Play », un son bref est joué.

B. Prérequis

Pour réaliser cet atelier, on doit disposer d'une machine virtuelle Linux Ubuntu ou bien avoir Ubuntu déjà installé sur une partition physique. Le compilateur GCC et le débogueur GDB ainsi que la version1.2 de la bibliothèque SDL doivent être installés.

C. Étapes à suivre

Dans cet atelier, on aura besoin d'un fichier main.c , fonction.h et fonction.c

C.I. Le fichier main.c

1. Édition du main.c : A l'aide de la commande suivante: gedit main.c (figure 1), on commence par éditer le fichier principale main.c

Figure 2. Édition du main.c avec gedit

2. **Déclaration des bibliothèques SDL** : Sur le fichier **main.c**, déclarer les entêtes suivants :

```
#include <SDL/SDL.h>
#include<SDL/SDL_image.h>
#include <SDL/SDL_mixer.h>
#include <SDL/SDL_ttf.h>
```

Figure 2. Déclaration des bibliothèques SDL

<SDL/SDL_image.h> Pour manipuler des images ayants des types autre que bmp
<SDL/SDL_ttf.h> Pour manipuler des textes
<SDL/SDL_mixer.h> Pour manipuler de l'audio.

3. **Déclaration des variables :**

On déclare les variables suivantes (figure 3)

screen : variable pointeur sur surface de l'arrière-plan
IMAGE, IMAGE_BTN1 et IMAGE_BTN2 : variables images des boutons
music et mus : variables pointeur sur les audios
txte : variable texte
event : variable pour l'évènement de l'utilisateur
boucle :un compteur pour la boucle de jeu.

```
SDL_Surface *screen;
image IMAGE, IMAGE_BTN1, IMAGE_BTN2;
Mix_Music *music;
Mix_Chunk *mus;
texte txte;
SDL_Event event;

int boucle=1;
```

Figure 3. Déclaration des variables

4. **Initialisation**

Avant de commencer la boucle de jeu (figure 4), on doit passer par l'étape de l'initialisation.

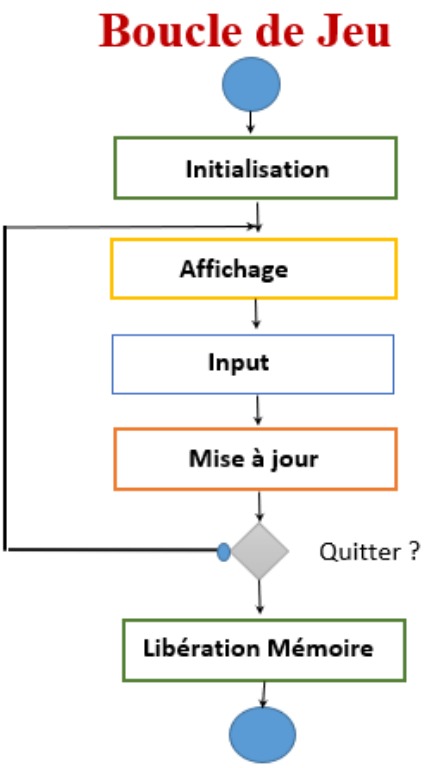

Figure 4. Boucle de jeu

On doit initialiser les composants graphiques, textuels et son en précisant leurs noms (figure 5).

```
if(SDL_Init(SDL_INIT_VIDEO|SDL_INIT_AUDIO|SDL_INIT_TIMER)==-1)
{
printf("Could not initialize SDL: %s.\n", SDL_GetError());
return -1;
}
```

Figure 5. Initialisation de la SDL

Ici on utilise des fonctions d'initialisation (figure 6) qu'on définira plus tard dans le fichier fonctions.c

```
initialiser_imageBACK(&IMAGE);
initialiser_imageBOUTON1(&IMAGE_BTN1);
initialiser_imageBOUTON2(&IMAGE_BTN2);
initialiser_audio(music);
initialiser_texte(&txte);
```

Figure 6. Initialisation des images, son et texte

5. **Définition de la boucle de jeu** : après l'initialisation, on doit placer toutes les étapes (affichage, lecture des input et la mise à jour des composants) à l'intérieur d'une boucle infinie appelée boucle de jeu où la variable *boucle* est toujours égale à 1 (figure 7).

```
while(boucle)
{

}
```

Figure 7. Boucle de jeu

6. **L'affichage :** à l'intérieur de la boucle de jeu (la boucle while), on affiche les composants graphiques du jeu, ici on utilise des fonctions qu'on définira plus tard dans le fichier fonctions.c (figure 8).

```
afficher_imageBMP(screen,IMAGE);
afficher_imageBTN1(screen, IMAGE_BTN1);
afficher_imageBTN2(screen, IMAGE_BTN2);
afficher_texte(screen, txte);
```

Figure 8. Affichage des images et du texte

7. **L'INPUT** : L'écoute des événements provenant des périphériques comme la souris ou le clavier
   On utilise une boucle while qui teste sur l'input de SDL_POLLEvent.
   SDL_Pollevent permet de lire l'événement sans bloquer la boucle du jeu.

```
while(SDL_PollEvent(&event))
{
}
```

Figure 9. Lecture des événements

8. **Détermination du type d'événement** : Selon le type d'événement donné par *event.type ,* on peut mettre à jour la boucle de jeu (figure 10).

```
switch(event.type)
{

}
```

Figure 10. Détermination du type d'événement

L'événement peut être :

- *event.type==SDL_KEYDOWN : appui sur une touche de clavier.*
- *event.type==SDL_MOUSEMOTION : un mouvement de la souris (figure 11)*

```
case SDL_MOUSEMOTION:
if(event.motion.y<=150 && event.motion.y>=100
&&(event.motion.x<=423 && event.motion.x>=213))
initialiser_audiobref(mus);
break;
```

Figure 11. Un mouvement de la souris

- *event.type==SDL_MOUSEBUTTONUP : un clic de la souris (figure 12)*

```
case SDL_MOUSEBUTTONDOWN:
if(event.button.button==SDL_BUTTON_LEFT
   && (event.motion.y<=300 && event.motion.y>=250
   && event.motion.x<=423 && event.motion.x>=213))
boucle=0;
break;
```

Figure 12. Un clic de la souris

- event.type==SDL_QUIT : appui sur le bouton fermer de la fenêtre, utiliser SDL_QUIT signifie arrêter tous les sous-systèmes SDL et libérer les ressources qui ont été allouées (figure 13)

```
case SDL_QUIT:
boucle=0;
break;
```

Figure 13. Quitter la fenêtre

9. **Mise à jour de l'écran :**
   La mise à jour de l'écran consiste à le rafraîchir à l'aide de **SDL_Flip**() à la fin de la boucle de jeu (figure 14).

```
SDL_Flip(screen);
```

Figure 14. Rafraîchir l'écran

10. **Libération des ressources utilisées puis quitter :** on libère les composants graphiques utilisés, ici on utilise des fonctions qu'on définira plus tard dans le fichier fonctions.c (figure 15) et on n'oublie pas de mettre SDL_QUIT à la fin.

```
liberer_image(IMAGE);
liberer_image(IMAGE_BTN1);
liberer_image(IMAGE_BTN2);
liberer_musique(music);
liberer_musiquebref(mus);
liberer_texte(txte);
```

Figure 15. Libération des ressources

Ci-dessous, le fichier main.c complet du menu:

main.c

```c
#include<stdio.h>
#include<stdlib.h>
#include <SDL/SDL.h>
#include<SDL/SDL_image.h>
#include <SDL/SDL_mixer.h>
#include <SDL/SDL_ttf.h>
#include "fonction.h"


int main()
{
//Declaration des variables
SDL_Surface *screen;
image IMAGE, IMAGE_BTN1, IMAGE_BTN2;
Mix_Music *music;
Mix_Chunk *mus;
texte txte;
SDL_Event event;

int boucle=1;


//initialisation de la  SDL
if(SDL_Init(SDL_INIT_VIDEO|SDL_INIT_AUDIO|SDL_INIT_TIMER)==-1)
{
printf("Could not initialize SDL: %s.\n", SDL_GetError());
return -1;
}
```

```c
//réglage de la fenetre
screen=SDL_SetVideoMode(SCREEN_W, SCREEN_H, 32,SDL_SWSURFACE|SDL_DOUBLEBUF);

//initialisation

initialiser_imageBACK(&IMAGE);
initialiser_imageBOUTON1(&IMAGE_BTN1);
initialiser_imageBOUTON2(&IMAGE_BTN2);
initialiser_audio(music);
initialiser_texte(&txte);

/*********************************/
//boucle du menu
/*********************************/
while(boucle)
{
//Affichage
afficher_imageBMP(screen,IMAGE);//pour afficher l'image du background
afficher_imageBTN1(screen, IMAGE_BTN1);//afficher le bouton play
afficher_imageBTN2(screen, IMAGE_BTN2);//afficher le bouton play
afficher_texte(screen, txte);
while(SDL_PollEvent(&event))
{
    switch(event.type)
    {
        case SDL_QUIT:
        boucle=0;
        break;
```

```
59          case SDL_MOUSEBUTTONDOWN://quitter le menu
60          if(event.button.button==SDL_BUTTON_LEFT && (event.motion.y<=300 && event.motion.y>=250
61                                                  && event.motion.x<=423 && event.motion.x>=213))
62          boucle=0;//quitter le menu
63          break;
64          case SDL_MOUSEMOTION://jouer un son bref
65          if(event.motion.y<=150 && event.motion.y>=100 &&(event.motion.x<=423 && event.motion.x>=213))
66          initialiser_audiobref(mus);//jouer un son bref
67          break;
68      }
69  }
70
71  SDL_Flip(screen);//raffraichir l'ecran
72  }
73  //liberer les surfaces
74  liberer_image(IMAGE);
75  liberer_image(IMAGE_BTN1);
76  liberer_image(IMAGE_BTN2);
77
78  //liberer la musique
79  liberer_musique(music);
80  liberer_musiquebref(mus);
81
82  //liberer texte
83  liberer_texte(txte);
84  //Quitter la SDL
85  SDL_Quit();
86  return 0;
87  }
```

C.II. Le fichier fonction.h

1. **Définition des structures :**

   On a besoin de définir une structure image (figure 16) contenant les attributs suivants :

- url : une chaine de caractères contenant l'emplacement du fichier image.
- pos_img_affiche : son type est SDL_Rect qui définit la position de l'image par rapport à l'écran
- pos_img_ecran : son type est SDL_Rect qui précise quelle partie de l'image doit être affichée
- *img : Déclaration d'un pointeur de surface pour manipuler l'image

```
typedef struct
{
    char *url;
    SDL_Rect pos_img_affiche;
        SDL_Rect pos_img_ecran;
    SDL_Surface *img;
}image;
```

Figure 16. La structure image

On a besoin de définir une structure texte (figure 17) contenant les attributs suivants :

- *txt : un pointeur sur SDL_Surface pour manipuler le texte.
- Pos_txt : préciser sa position de l'image par rapport à l'écran
- Color_txt : couleur du texte en format (r,g,b) doit être entre 0 et 255
- *police : la police du texte se trouve sur un fichier de police (format.ttf)

```
typedef struct
{
    SDL_Surface *txt;
    SDL_Rect pos_txt;
    SDL_Colour color_txt;
    TTF_Font *police;
}texte;
```

Figure 17. La structure texte

2. **Définition des entêtes des fonctions :**
   - On définit pour la manipulation des images les fonctions d'initialisation, d'affichage et de libération (figure 18).

```
//image
void initialiser_imageBACK(image *imge);
void initialiser_imageBOUTON1(image *imgbtn);
void initialiser_imageBOUTON2(image *imgbtn);
void afficher_imageBMP(SDL_Surface *screen, image imge);
void afficher_imageBTN1(SDL_Surface *screen, image imge);
void afficher_imageBTN2(SDL_Surface *screen, image imge);
void liberer_image(image imge);
```

Figure 18. Entêtes des fonctions de manipulations des images

   - On définit pour la manipulation des audios, les fonctions d'initialisation, d'affichage et de libération (figure 19).

```
//audio
void initialiser_audio(Mix_Music *music);
void liberer_musique(Mix_Music *music);
void initialiser_audiobref(Mix_Chunk *music);
void liberer_musiquebref(Mix_Chunk *music);
```

Figure 19. Entêtes des fonctions de manipulations des audios

   - On définit pour la manipulation des textes les fonctions d'initialisation, d'affichage et de libération (figure 20).

```
void initialiser_texte(texte *txte);
void afficher_texte(SDL_Surface *screen, texte txte);
void liberer_texte(texte txte);
```

Figure 20. Entêtes des fonctions de manipulation des textes

Ci-dessous, le fichier fonction.h complet du menu:

here | fonction.h

```c
#ifndef FONCTION_H
#define FONCTION_H
#include<SDL/SDL.h>
#include <SDL/SDL_image.h>
#include <SDL/SDL_mixer.h>
#include <SDL/SDL_ttf.h>
#define SCREEN_H 410
#define SCREEN_W 640
typedef struct
{
    char *url;
    SDL_Rect pos_img_affiche;//partie de l'image qui doit etre affichée
        SDL_Rect pos_img_ecran; // position de l'image par rapport l'écran
    SDL_Surface *img;
}image;

typedef struct
{
    SDL_Surface *txt;
    SDL_Rect pos_txt;
    SDL_Colour color_txt;
    TTF_Font *police;
}texte;

//image
void initialiser_imageBACK(image *imge);
void initialiser_imageBOUTON1(image *imgbtn);
void initialiser_imageBOUTON2(image *imgbtn);

void afficher_imageBMP(SDL_Surface *screen, image imge);
void afficher_imageBTN1(SDL_Surface *screen, image imge);
void afficher_imageBTN2(SDL_Surface *screen, image imge);
void liberer_image(image imge);

//audio
void initialiser_audio(Mix_Music *music);
void liberer_musique(Mix_Music *music);

void initialiser_audiobref(Mix_Chunk *music);
void liberer_musiquebref(Mix_Chunk *music);
//texte
void initialiser_texte(texte *txte);
void afficher_texte(SDL_Surface *screen, texte txte);
void liberer_texte(texte txte);

#endif
```

C.III. Le fichier fonction.c

Dans ce fichier, toutes les fonctions dont les en têtes ont été déclarées dans le fichier fonction.h doivent être développées. On va commencer par les fonctions d'initialisation.

1. **Les fonctions d'initialisation** :

1.1. Initialisation des images :

- Dans la figure 21, on définit la fonction d'initialisation de l'image du background. On affecte à l'attribut url l'adresse de l'emplacement de l'image. On précise la position du background par rapport à l'écran qui est toujours (0,0). On affiche ici la totalité de l'image du background. On affiche la partie qui couvre tout l'écran.

```
void initialiser_imageBACK(image *imge)
{


imge->url="final.bmp";
imge->img=SDL_LoadBMP(imge->url);
if (imge->img == NULL){
    printf("unable to load background image %s \n",SDL_GetError());
    return ;}
imge->pos_img_ecran.x=0;
imge->pos_img_ecran.y=0;
imge->pos_img_affiche.x=0;
imge->pos_img_affiche.y=0;
imge->pos_img_affiche.h=SCREEN_H;
imge->pos_img_affiche.w=SCREEN_W;

}
```

Figure 21. La fonction d'initialisation de l'image du background

- Pour la fonction d'initialisation de l'image du bouton, on définit l'url de l'emplacement du fichier de l'image (figure 22)

```
imgbtn->url="play.png";
```

Figure 22. Emplacement du fichier image de bouton

- On initialise la position du premier deux boutons (figure 23) au milieu de l'écran sur l'axe des x (à l'horizontale) et au niveau de 1/3 de l'écran sur l'axe y (à la verticale).

```
imgbtn->pos_img_ecran.x=((SCREEN_W-imgbtn->pos_img_affiche.w)/2);
imgbtn->pos_img_ecran.y=((SCREEN_H-imgbtn->pos_img_affiche.h)/3);
```

Figure 23. Initialisation de l'emplacement du premier bouton

- Le deuxième bouton sera placé au milieu de l'écran sur l'axe des x (à l'horizontale) et au niveau de 2/3 de l'écran sur l'axe y (à la verticale).

1.2. Initialisation du texte :

- Pour initialiser le texte, on utilise TTF_Init() de SDL_ttf (figure 24). On charge la police ubuntu-B et on initialise la taille à 45. On initialise la couleur du texte à (0,0,0) puis on initialise la position du texte.
-

```
void initialiser_texte(texte *txte)
{
TTF_Init();
txte->police = TTF_OpenFont("Ubuntu-B.ttf", 45);
txte->color_txt.r=0;
txte->color_txt.g=0;
txte->color_txt.b=0;

txte->pos_txt.x=210;
txte->pos_txt.y=0;
}
```

Figure 24. Initialisation du texte

1.3. Initialisation de l'audio : Pour initialiser l'audio, on utilise la fonction de SDL_mixer : MixOpenAudio (figure 25).

```
if(Mix_OpenAudio(44100,MIX_DEFAULT_FORMAT,MIX_DEFAULT_CHANNELS,1024)==-1){
    printf("%s",SDL_GetError());
```

Figure 25.Initialisation de l'audio

2. Les fonctions d'affichage :

2.1. L'affichage de l'image du background : elle se fait à travers la fonction SDL_BlitSurface permettant de coller l'image sur un écran (figure 26).

```
SDL_BlitSurface(imge.img, &imge.pos_img_affiche, screen, &imge.pos_img_ecran);
```

Figure 26. Collage de l'image du background sur l'écran

2.2. L'affichage du bouton quitter

```
SDL_BlitSurface(imge.img, NULL, screen, &imge.pos_img_ecran);
```

Figure 27. Collage de l'image du bouton quitter sur l'écran

2.3. Affichage du texte : pour écrire le texte sur l'image du background (blitter sur un fond non uni), on utilise ici TTF_RenderText_Blended puis SDL_BlitSurface (figure 28)

```
void afficher_texte(SDL_Surface *screen, texte txte)
{
txte.txt=TTF_RenderText_Blended(txte.police, "Bienvenue",txte.color_txt);
SDL_BlitSurface (txte.txt, NULL, screen, &txte.pos_txt);
}
```

Figure 28. Affichage du texte

2.4. Jouer le son : pour charger la musique, on utilise Mix_LoadMUS. Pour jouer le son, on utilise Mix_PlayMusic. Et pour régler le volume, on utilise Mix_VolumeMusic (figure 29)

```
music=Mix_LoadMUS("CODEX Installer Music .mp3");
Mix_PlayMusic(music,-1);
Mix_VolumeMusic(MIX_MAX_VOLUME/3.5);
```

Figure 29. Chargement et play du son

3. Libération des ressources :
   3.1. Libération des images : on utilise SDL_FreeSurface (figure 30).

```
void liberer_image(image imge)
{
SDL_FreeSurface(imge.img);
}
```

Figure 30. Libération des images

   3.2. Libération du texte : on utilise TTF_CloseFont pour fermer la police puis TTF_Quit pour arrêter le SDL_ttf (figure 31).

```
void liberer_texte(texte txte)
{
TTF_CloseFont (txte.police);
TTF_Quit();
}
```

Figure 31. Libération du texte

   3.3. Libération de l'audio : On utilise la fonction Mix_FreeChunk (figure 32).

```
void liberer_musiquebref(Mix_Chunk *music)
{
Mix_FreeChunk(music);
}
```

Figure 32. Libération de l'audio

Ci-dessous, le fichier fonction.c complet :

t here | fonction.c

```
1  #include <SDL/SDL.h>
2  #include <SDL/SDL_image.h>
3  #include <SDL/SDL_mixer.h>
4  #include <SDL/SDL_ttf.h>
5  #include "fonction.h"
6
7
8  /*********************************
9  *******TRAITEMENT DES IMAGES*******/
10
11 void initialiser_imageBACK(image *imge)
12 {
13
14 //chargement de l'image
15 imge->url="final.bmp";
16 imge->img=SDL_LoadBMP(imge->url);
17 if (imge->img == NULL){
18     printf("unable to load background image %s \n",SDL_GetError());
19     return ;}
20 imge->pos_img_ecran.x=0;
21 imge->pos_img_ecran.y=0;
22 imge->pos_img_affiche.x=0;
23 imge->pos_img_affiche.y=0;
24 imge->pos_img_affiche.h=SCREEN_H;
25 imge->pos_img_affiche.w=SCREEN_W;
26
27 }
```

```
30  void initialiser_imageBOUTON1(image *imgbtn)
31  {
32  //chargement de l'image
33  imgbtn->url="play.png";
34  imgbtn->img=IMG_Load(imgbtn->url);
35  if (imgbtn->img == NULL){
36      printf("unable to load background image %s \n",SDL_GetError());
37      return ;}
38  imgbtn->pos_img_affiche.x=0;
39  imgbtn->pos_img_affiche.y=0;
40  imgbtn->pos_img_affiche.w=237;
41  imgbtn->pos_img_affiche.h=58;
42  imgbtn->pos_img_ecran.x=((SCREEN_W-imgbtn->pos_img_affiche.w)/2);
43  imgbtn->pos_img_ecran.y=((SCREEN_H-imgbtn->pos_img_affiche.h)/3);
44
45  }
```

```
46  void initialiser_imageBOUTON2(image *imgbtn)
47  {
48  //chargement de l'image
49  imgbtn->url="quit.png";
50  imgbtn->img=IMG_Load(imgbtn->url);
51  if (imgbtn->img == NULL){
52      printf("unable to load background image %s \n",SDL_GetError());
53      return ;}
54
55  imgbtn->pos_img_affiche.x=0;
56  imgbtn->pos_img_affiche.y=0;
57  imgbtn->pos_img_affiche.w=236;
58  imgbtn->pos_img_affiche.h=55;
59  imgbtn->pos_img_ecran.x=((SCREEN_W-imgbtn->pos_img_affiche.w)/2);
60  imgbtn->pos_img_ecran.y=(2*(SCREEN_H-imgbtn->pos_img_affiche.h)/3);
61
62  }
63  void afficher_imageBMP(SDL_Surface *screen, image imge)
64  {
65  //Coller et Afficher l'image du back dans screen
66  SDL_BlitSurface(imge.img, &imge.pos_img_affiche, screen, &imge.pos_img_ecran);
67
68  }
69  void afficher_imageBTN1(SDL_Surface *screen, image imge)
70  {
71  //Coller et Afficher l'image du btn play dans screen
72  SDL_BlitSurface(imge.img, NULL, screen, &imge.pos_img_ecran);
73  }
```

```
75  void afficher_imageBTN2(SDL_Surface *screen, image imge)
76  {
77  //Coller et Afficher l'image du btn quit dans screen
78  SDL_BlitSurface(imge.img, NULL, screen, &imge.pos_img_ecran);
79
80  }
81  void liberer_image(image imge)
82  {
83  SDL_FreeSurface(imge.img);
84  }
85
86  /************************************
87  *******TRAITEMENT MUSIQUE*******/
88
89  void initialiser_audio(Mix_Music *music)
90  {
91  //initialiser les fonction audio de SDL_mixer
92  if(Mix_OpenAudio(44100,MIX_DEFAULT_FORMAT,MIX_DEFAULT_CHANNELS,1024)==-1){
93      printf("%s",SDL_GetError());
94
95  }
96  music=Mix_LoadMUS("CODEX Installer Music .mp3");//chargement de la musique
97  Mix_PlayMusic(music,-1);//jouer la musique
98  Mix_VolumeMusic(MIX_MAX_VOLUME/3.5);
99
100 }
101
```

```
101  void liberer_musique(Mix_Music *music)
102  {
103  Mix_FreeMusic(music);
104  }
105  void initialiser_audiobref(Mix_Chunk *music)
106  {
107  Mix_OpenAudio(44100, MIX_DEFAULT_FORMAT, 2, 2048);
108  music = Mix_LoadWAV("simple.wav");
109  Mix_PlayChannel( -1, music, 0 );
110  if(music==NULL) printf("%s",SDL_GetError());
111  }
112  void liberer_musiquebref(Mix_Chunk *music)
113  {
114  Mix_FreeChunk(music);
115  }
116  /**********************************
117  *******TRAITEMENT TEXTE*******/
118  void initialiser_texte(texte *txte)
119  {
120  TTF_Init();//initialiser SDL_ttf
121  txte->police = TTF_OpenFont("Ubuntu-B.ttf", 45);//chargement d'une police et initialiser la taille
122  txte->color_txt.r=0;
123  txte->color_txt.g=0;
124  txte->color_txt.b=0;
125
126  txte->pos_txt.x=210;
127  txte->pos_txt.y=0;
128  }
```

```
130  void afficher_texte(SDL_Surface *screen, texte txte)
131  {
132  txte.txt=TTF_RenderText_Blended(txte.police, "Bienvenue",txte.color_txt);//écrire le message "Bienvenu"
133  SDL_BlitSurface (txte.txt, NULL, screen, &txte.pos_txt);//coller la surface
134  }
135
136  void liberer_texte(texte txte)
137  {
138  TTF_CloseFont (txte.police);//fermer la police
139  TTF_Quit();//arreter la SDL_ttf
140  }
```