Projet 1A

Ben Mahmoud Landolsi Houssem

2020

Objectives

- Gérer un projet
  - Mise en application de ce qu’on a appris dans la programmation C

Gérer un projet

Définition et organisation des tâches
Manipulations des outils de travail

Agenda

1. De quoi s’agit-il ?
2. Prototype du jeu Code
3. Outils de travail
4. Démonstration

1. De quoi s’agit-il ?

Création d’un jeu vidéo 2D

Nom: Mario World

Personnages: Mario / Luigi

Environnement: Scène sombre de bataille

Musique: Musique du Fond(début) /externe(Fin)

2. Prototype du jeu

Initialisation du jeu

Affichage du jeu(scène+texte+personnage)

Déplacement de 2 entités(Mario/luigi)

Collision bounding box

Initialisation du jeu

Déclaration des variables
L’initiation de la bibliothèque SDL
Initialisation de la vidéo
Initialisation et chargement du son
Initialisation des événements

Initialisation du jeu

```c
#include "SDL/SDL.h"
#include <SDL/SDL_mixer.h>
#include <SDL/SDL_ttf.h>
#include "fonctions.h"

 SDL_Surface *screen=NULL; //reference the backbuffer
 Mix_Music *music;//Contruct Mix_Music pointer
 Mix_Music *musicgameover;
 SDL_Surface *mario=NULL; //reference our image
 SDL_Surface *luigi=NULL;
 SDL_Rect positionmario;
 SDL_Rect positionluigi;//rect to decribe the source destination region of out blit
 SDL_Rect positiontext;
 SDL_Rect positiontextgameover;
 SDL_Event event;
 TTF_Font *police = NULL;
 TTF_Font *policegameover=NULL;
 SDL_Surface *text;
 SDL_Surface *textgameover;
 int gameover=0;

```

La fonction initialiser()

```c
int initialiser()
{

    if(SDL_Init(SDL_INIT_VIDEO|SDL_INIT_AUDIO)!=0)
    {
        printf("unable to initialize SDL: %s\n",SDL_GetError());
        return 1;
    }

    if(TTF_Init()== -1)
    {
        printf("Erreur d'initialisation de TTF_Init : %s\n", TTF_GetError());
        return 1;
    }


    police = TTF_OpenFont("signature.ttf", 65);

    policegameover= TTF_OpenFont("gameover.ttf",50);

    SDL_Color couleurrouge = {250, 0, 0};

    text = TTF_RenderText_Blended(police,"Mario World",couleurrouge);

    textgameover = TTF_RenderText_Blended(policegameover,"Game Over",couleurrouge);
```

La fonction initialiser()

```
//API Mixer Initialization
if(Mix_OpenAudio(44100,MIX_DEFAULT_FORMAT,MIX_DEFAULT_CHANNELS,1024)==-1)
{
    printf("%s",Mix_GetError());
}


music = Mix_LoadMUS("smbss_intro-cartoon.mp3");//load the music
Mix_PlayMusic(music,-1);//play music forever("-1")

musicgameover = Mix_LoadMUS("smb_gameover.wav");//load the music


screen = SDL_SetVideoMode( 600, 400, 16, SDL_HWSURFACE|SDL_DOUBLEBUF);
SDL_WM_SetCaption("Gestion des évènements en SDL", NULL);

if (screen==NULL)
{
    printf("Unable to set video mode : %s\n",SDL_GetError());
    return 1;
}
```

La fonction initialiser()

```
//Load the bitmap into the image sureface,and check for success
mario=SDL_LoadBMP("Mario.bmp"); //you can use IMG_load

if (mario==NULL)
{
    printf("Unable to load bitmap:%s\n",SDL_GetError());
    return 1;
}

 luigi=SDL_LoadBMP("luigi.bmp"); //you can use IMG_load

if (luigi==NULL)
{
    printf("Unable to load bitmap:%s\n",SDL_GetError());
    return 1;
}

//Construct the source rectangle for our blit
positionmario.x=0;
positionmario.y=200;
positionmario.w=mario->w;
positionmario.h=mario->h;

positionluigi.x=screen->w-luigi->w;
positionluigi.y=200;
positionluigi.w=luigi->w;
positionluigi.h=luigi->h;
```

```
    positiontextgameover.x=150;
    positiontextgameover.y=150;

    SDL_EnableKeyRepeat(10, 10); /* Activation de la répétition des touches */
}
```

Affichage du jeu

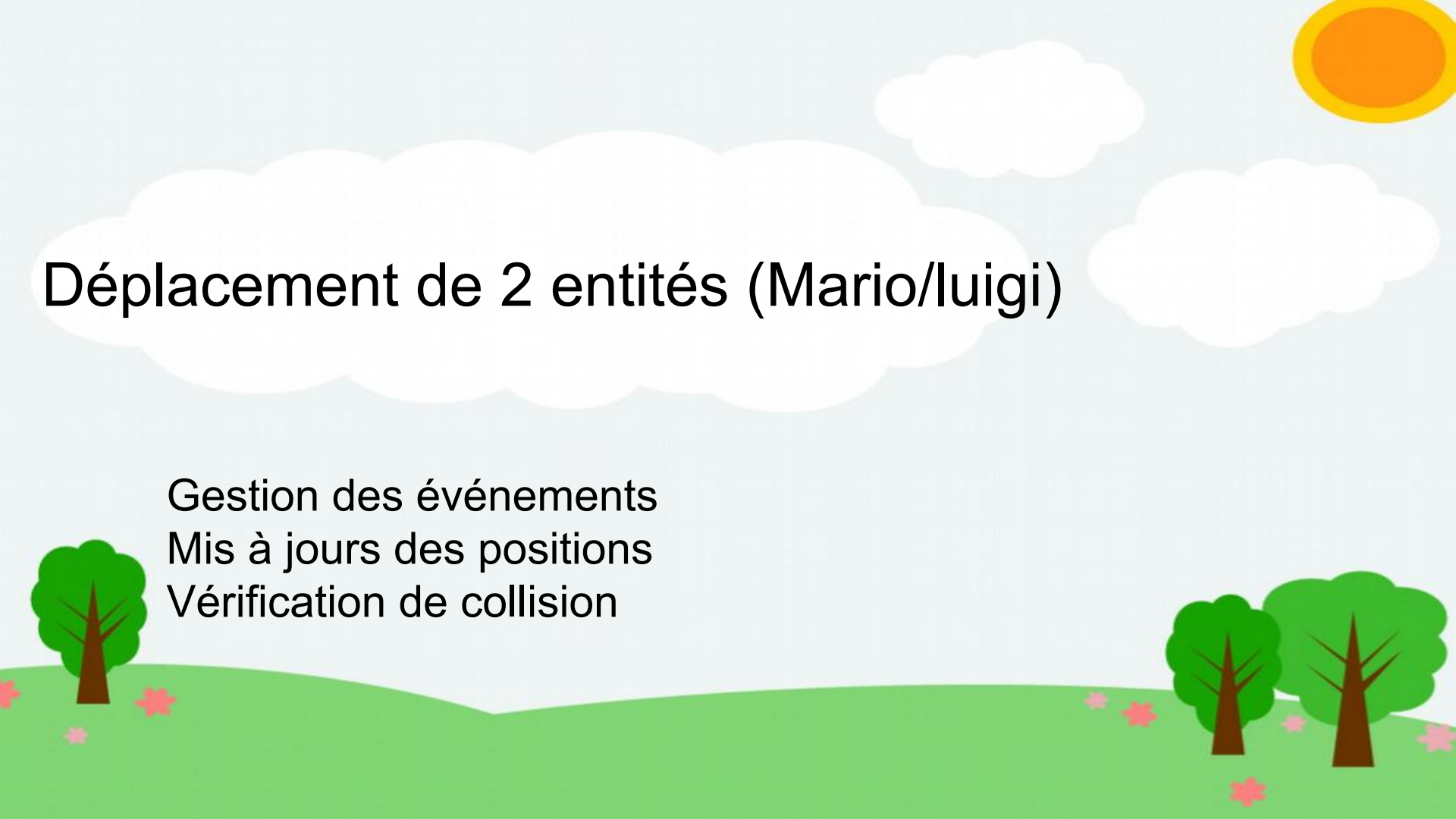
Mis à jour des images sur l’écran

La fonction afficher()

```
void afficher()
{
    SDL_FillRect(screen, NULL, SDL_MapRGB(screen->format, 0, 0, 0));

    SDL_BlitSurface(text, NULL, screen, &positiontext);

    //Blit the image to the backbuffer
    SDL_BlitSurface(mario, NULL, screen, &positionmario);

    SDL_BlitSurface(luigi, NULL, screen, &positionluigi);
    if(gameover==1)
    {
        SDL_BlitSurface(textgameover, NULL, screen, &positiontextgameover);
    }
    //flip the backbuffer to the primary Hardware Video Memory
    SDL_Flip(screen);

}
```

Déplacement de 2 entités (Mario/luigi)

Gestion des événements
Mis à jours des positions
Vérification de collision



Déplacement de 2 entités (Mario/luigi)

```
int deplacer()
{
     int done=0;
     while (SDL_PollEvent(&event))
     {    //check for messages
          switch(event.type)
          {
               case SDL_QUIT:
                    done = 1;
                    printf("Event SDLquit!\n");
                    break;
                    //check for keypresses
               case SDL_KEYDOWN:
                    printf("Event SDL KeyDown!\n");
                    switch(event.key.keysym.sym)
                    {
                         case SDLK_ESCAPE:
                              done=1;
                              printf("Event SDLESCAPE!\n");
                              break;
                         case SDLK_UP:
                              if(gameover==1)
                              {
                                   break;
                              }
                              positionmario.y--;
                              printf("Event SDLK UP !\n");
                              break;
```

Déplacement de 2 entités (Mario/luigi)

```
case SDLK_DOWN:
    if(gameover==1)
    {
        break;
    }
    positionmario.y++;
    printf("Event SDLK DOWN!\n");
    break;
case SDLK_RIGHT:
    if(gameover==1)
    {
        break;
    }
    positionmario.x++;
    printf("Event SDLK RIGHT!\n");
    break;
case SDLK_LEFT:
    if(gameover==1)
    {
        break;
    }
    positionmario.x--;
    printf("Event SDLK LEFT!\n");
    break;
```

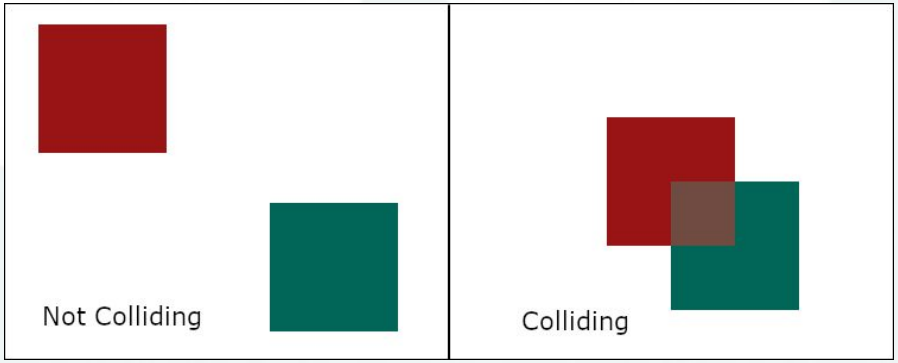
Déplacement de 2 entités (Mario/luigi)

```
    //Deplacement luigi.
case SDLK_z:
     if(gameover==1)
     {
          break;
     }
     positionluigi.y--;
     printf("Event SDLK UP !\n");
     break;
case SDLK_s:
     if(gameover==1)
     {
          break;
     }
     positionluigi.y++;
     printf("Event SDLK DOWN!\n");
     break;
case SDLK_d:
     if(gameover==1)
     {
          break;
     }
     positionluigi.x++;
     printf("Event SDLK RIGHT!\n");
     break;
case SDLK_q:
     if(gameover==1)
     {
          break;
```

Déplacement de 2 entités(Mario/luigi)

```
                    positionluigi.x--;
                    printf("Event SDLK LEFT!\n");
                    break;
                }
                break;
            }

            if(collision(positionmario,positionluigi)==0)
            {
                printf("Il n y a pas de collision entre mario et luigi");

            }
            else
            {
                printf("I y a collision entre mario et luigi");

                gameover=1;

                Mix_HaltMusic();

                Mix_PlayMusic(musicgameover,0);
            }
        }
        return done;
}
```

Collision bounding box



Détection des collisions
Vérification de l’intersection des rectangles

Collision bounding box

```
/*
Implementation de la fonction collision .
*/
int collision(SDL_Rect p1, SDL_Rect p2)
{
    if(p2.y+p2.h<p1.y || p1.y+p1.h<p2.y||p1.x+p1.w<p2.x || p2.x+p2.w<p1.x)
    {
        return 0;//pas de collision.
    }
    else
    {
        return 1;//il y a collision.
    }

};
```

3. Outils de travail

Système d’exploitation Linux(Ubuntu)
Terminal Ubuntu+outils de développements
Programmation en C
Bibliothèque SDL
Git-Hub

Programmation en C

Implémentation du code du jeu avec la programmation en C

Le code va contenir trois fichiers importants: Main.c, Fonctions.h et Fonctions.c

Main.c

Déclaration des variables

L’appel des fonctions dans la boucle du jeu

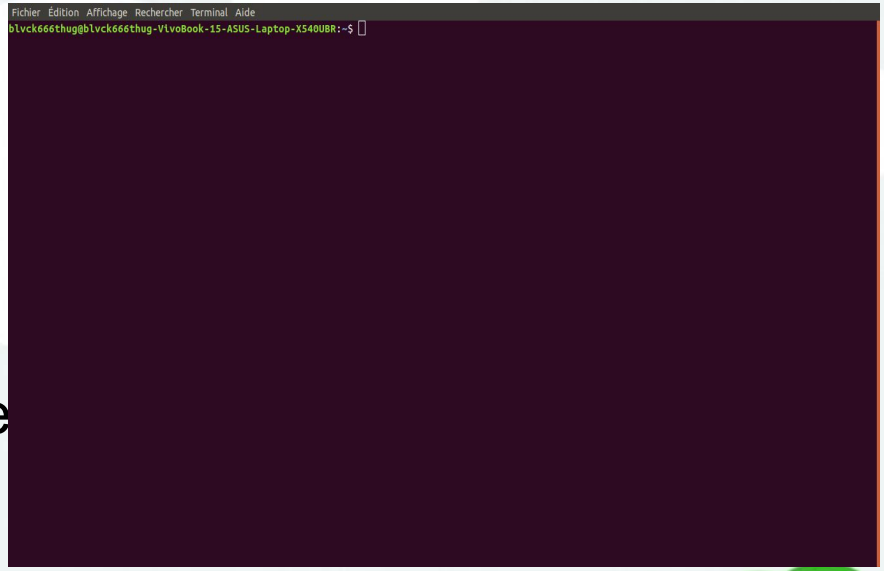
Fonctions.h

Les structures et les entêtes des fonctions

Fonctions.c

l'implémentation des fonctions

Terminal Ubuntu



Un Programme qui émule une console dans une interface graphique

Il permet de lancer des commandes pour manipuler notre ordinateur

Outils de développement Terminal

GCC (GNU Compiler Collection)
Make (Gestionnaire de Compilation Make)
Git-Hub

GCC (GNU Compiler Collection)

Une suite de logiciels libres de compilation utilisé dans le monde Linux dès qu'on veut transcrire du code source en langage machine

Make (Gestionnaire de Compilation Make)

Automatiser la phase de compilation.

l ‘élimination de fichiers temporaires créés par votre éditeur de texte et le compilateur.

Bibliothèque SDL

Simple Direct Media Layer, bibliothèque de développement multi-plateformes qui fournir un accès au matériel audio, clavier, souris, manettede jeu, etc

Elle permet de développer des programmes gérant le son, la vidéo, le clavier, la souris ,etc

Git-Hub

Définition

Un service web d'hébergement et de gestion de développement de logiciels

Info P1A sur Git

Code Source

https://github.com/BLVCK666Thug/MarioWorld

Documentation

https://github.com/BLVCK666Thug/MarioWorld/wiki/Documentation-du-code

4.Démonstration du jeu

Début du jeu
Déplacement de Mario et Luigi
Fin du jeu
Video

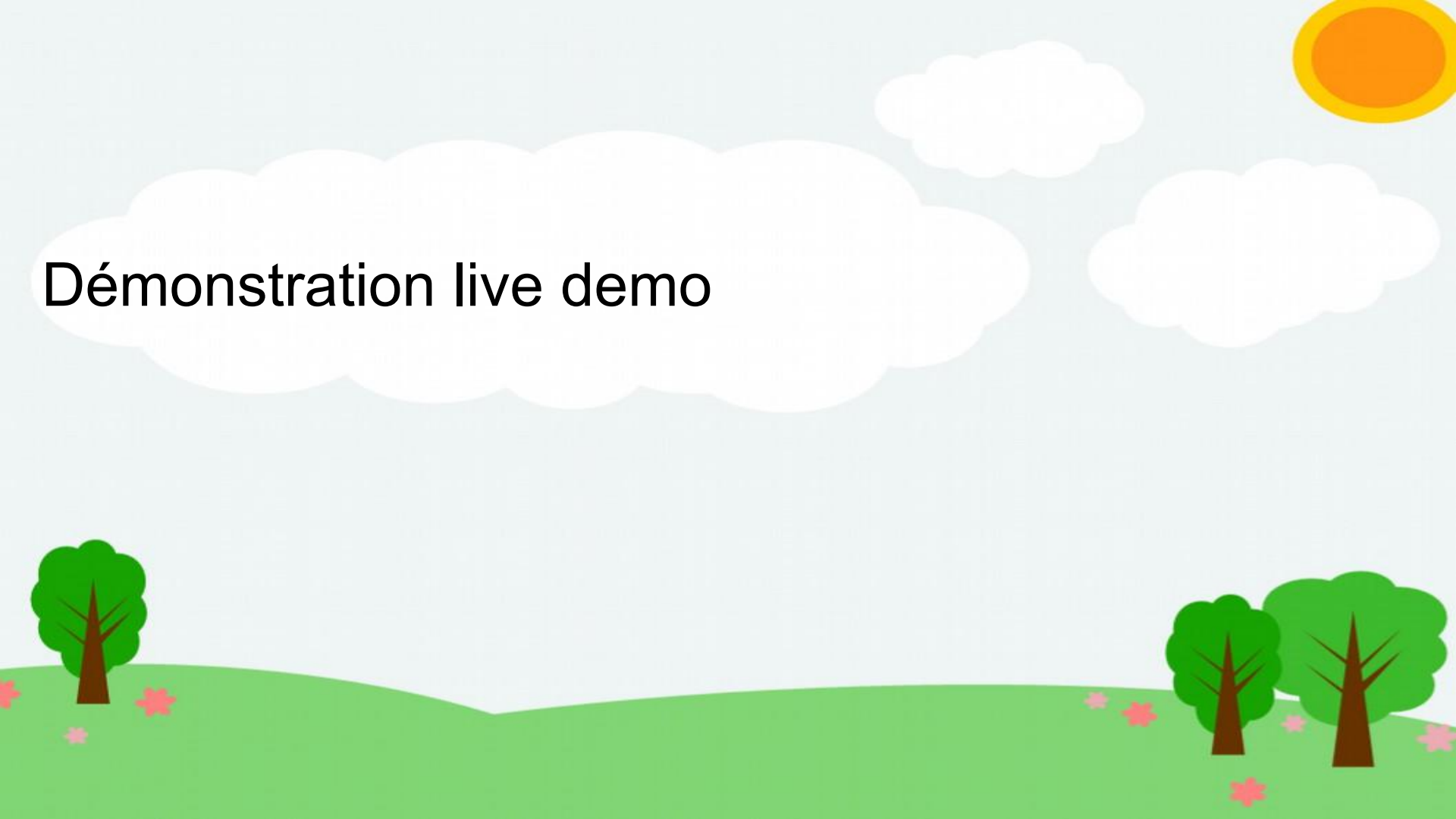
Début du jeu


blvck666thug@blvck666thug-VivoBook-15-ASUS-Laptop-X540UBR: ~/Bureau/ESPRIT/Prog/S...
Gestion des évènements en SDL
Mario World

Déplacement de Mario et Luigi

Fin du jeu

Démonstration live demo



Perspectives

Background
Obstacles
Animation
Déplacement avec accéleration et Saut

Merci pour votre attention