

TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

**REPORT ON MINI PROJECT IN C**

**Submitted by: Submitted to:**

Rajesh Neupane 076BCT055 Department of Computer Engineering

Rishav Bhattarai 076BCT058

Pratik Dahal 076BCT051

Rahul Kumar Jha 076BCT053

**Mini project in C:**

**Flappy Bird:**

This C mini project is about a mini game, named as Flappy bird. Although being a small game, it’s designing and programming required a lot of C features, since its coded in C language. Structures, pointers, basic header files and libraries for gaming prospects are used here rapidly.

**Salient featured of Flappy bird:**

* You can fly your bird with a click(gap button) and overpass the obstacles(i.e. walls).
* You can give your bird a push in either front or back direction by clicking side buttons.
* You can play this game for infinite time, i.e. there’s no time limitation or ‘game over’ thing here. So, you can fly and fly and fly with your bird forever.
* Simple form of graphics are used to reduce loading time.
* It can enhance your reacting and focusing skills.

**At the end…**

Although this game looks so simple, it required a lot of hardwork and creative ideas to build it. Uses of so many in-built functions and other features of C were required to give this game’s vision a development. We hope you like it and try this game once.

**Source code:**

#include<SDL2/SDL.h>

#include<stdio.h>

#include<SDL2/SDL\_ttf.h>

#include"main.h"

#include<SDL2/SDL\_mixer.h>

#define GRAVITY 0.18f

void loadGame(GameState \*game){

SDL\_Surface \*Surface=NULL;

//load images and create rendering texture from them

Surface =IMG\_Load("pebbles.JPG");

if (Surface==NULL)

{ printf("cannot find pebbles.jPEG !\n\n");

SDL\_Quit();

exit(1);

}

game->ground=SDL\_CreateTextureFromSurface(game->renderer,Surface);

SDL\_FreeSurface(Surface);//free the memory the surface was using

Surface =IMG\_Load("walls.JPG");

if (Surface==NULL)

{ printf("cannot find walls.JPEG !\n\n");

SDL\_Quit();

exit(1);

}

game->wall=SDL\_CreateTextureFromSurface(game->renderer,Surface);

SDL\_FreeSurface(Surface);//free the memory the surface was using

//init walls

for (int i=0;i<500;i++)

{

game->walls[i].x=400+i\*200;

game->walls[i].y1=0;

int k= rand()%250;

game->walls[i].y2=k+150;

game->walls[i].w=60;

game->walls[i].h1=20+k;

game->walls[i].h2=250;

}

Surface =IMG\_Load("bird.JPG");

if (Surface==NULL)

{ printf("cannot find bird.JPEG!\n\n");

SDL\_Quit();

exit(1);

}

game->bird1=SDL\_CreateTextureFromSurface(game->renderer,Surface);

SDL\_FreeSurface(Surface);//free the memory the surface was using

Surface =IMG\_Load("birdh.JPG");

if (Surface==NULL)

{ printf("cannot find bird.JPEG!\n\n");

SDL\_Quit();

exit(1);

}

game->bird2=SDL\_CreateTextureFromSurface(game->renderer,Surface);

SDL\_FreeSurface(Surface);//free the memory the surface was using

//load sounds

game->bgMusic =Mix\_LoadMUS("bgmusic.WAV");

SDL\_FreeSurface(Surface);

game->fbird.x =180;

game->fbird.y =220;

game->fbird.dy=0;

}

void process(GameState \*game)

{

game->time++;

Bird \*fbird=&game->fbird;

fbird->y +=fbird->dy;

if (game->fbird.jump==true){

fbird->dy +=GRAVITY;

fbird->x +=2;

}

if (game->time%30==0){

if (game->fbird.animframe==0)

{

game->fbird.animframe=1;

}

else {game->fbird.animframe=0;}

}

}

void collisionDetect(GameState \*game)

{

for (int i=0;i<500;i++){

float bw=36,bh=24;

float bx=game->fbird.x,by=game->fbird.y;

float gx=game->walls[i].x,gy2=game->walls[i].y2,gy1=0,gw=game->walls[i].w,gh1=game->walls[i].h1,gh2=game->walls[i].h2;

if (bx<0){

game->fbird.x=0;

game->fbird.dy=0;

}

if (by<0){

game->fbird.y=0;

//stop any jump velocity

game->fbird.dy=0;

}

//check collision with the above wall

if(by>=0 && by<=gh1)

{

if (bx+bw>=gx && bx+bw<=gx+gw){

game->fbird.x=gx-bw;

bx =gx-bw;

game->fbird.dy=0;

}

else if( bx<=gx+gw && bx>=gx){

game->fbird.x=gx+gw;

bx =gx+gw;

game->fbird.dy=0;

}

}

if(by+bh<gy2&&by>gh1)

{

if (by+bh>gy2){

game->fbird.y=gy2-bh;

by=gy2-bh;

game->fbird.dy=0;

}

if (by<gh1){

game->fbird.y=gh1;

by=gh1;

game->fbird.dy=0;

}

}

if(by+bh>=400){

game->fbird.y=400-24;

by=400-24;

game->fbird.dy=0;

}

if (by>=gy2 && by+bh<=400){

if (bx+bw>=gx && bx+bw<=gx+gw){

game->fbird.x=gx-bw;

bx =gx-bw;

game->fbird.dy=0;

}

else if( bx<=gx+gw && bx>=gx){

game->fbird.x=gx+gw;

bx =gx+gw;

game->fbird.dy=0;

}

}

}

}

void doRender(SDL\_Renderer \*renderer,GameState \*game){

//render display

//set the drawing color to blue

SDL\_SetRenderDrawColor(renderer,144,234,248,255);

//clear the screen to the above given color

SDL\_RenderClear(renderer);

if(game->fbird.animframe ==1){

SDL\_Rect rect ={game->scrollx+game->fbird.x,game->fbird.y,36,24};

SDL\_RenderCopyEx(renderer,game->bird1,NULL,&rect,0,NULL,0);}

if (game->fbird.animframe==0){

SDL\_Rect rect1 ={game->scrollx+game->fbird.x,game->fbird.y,36,24};

SDL\_RenderCopyEx(renderer,game->bird2,NULL,&rect1,0,NULL,0);}

//draw the wall image

for (int i=0;i<500;i++){

int h =(game->walls[i].x),k=(game->walls[i].y2);

int h1 =game->walls[i].h1;

SDL\_Rect wallRect={game->scrollx+h,0,60,h1};

SDL\_RenderCopy(renderer,game->wall,NULL,&wallRect);

SDL\_Rect wall2Rect={game->scrollx+h,k,60,250};

SDL\_RenderCopy(renderer,game->wall,NULL,&wall2Rect);

//we are done drawing "present" to the screen what we've drawn

}

//draw the ground image

SDL\_Rect groundRect={0,400,640,80};

SDL\_RenderCopy(renderer,game->ground,NULL,&groundRect);

SDL\_RenderPresent(renderer);

// //draw the fonts in the screen

// SDL\_Rect fontRect={200,60,48,36};

// SDL\_RenderCopy(renderer,game->label,NULL,&fontRect);

}

int processEvents(SDL\_Window \*window,GameState \*game){

int done=0;

SDL\_Event event;

//check for events

while (SDL\_PollEvent(&event))

{

switch(event.type)

{

case SDL\_WINDOWEVENT\_CLOSE:

{

if (window)

{

SDL\_DestroyWindow(window);

window =NULL;

done=1;

}

}

break;

case SDL\_KEYDOWN:

{

switch(event.key.keysym.sym)

{

case SDLK\_ESCAPE:

done =1;

break;

case SDLK\_SPACE:

game->fbird.jump=true;

game->fbird.dy =-5;

}

break;

}

case SDL\_QUIT :

//quit out of the game

done =1;

break;

}

}

const Uint8 \*state = SDL\_GetKeyboardState(NULL);

if(state[SDL\_SCANCODE\_LEFT]){

game->fbird.x -=10;

}

if(state[SDL\_SCANCODE\_RIGHT]){

game->fbird.x +=10;

}

// if(state[SDL\_SCANCODE\_UP]){

// game->fbird.y -=10;

// }

// if(state[SDL\_SCANCODE\_DOWN]){

// game->fbird.y +=10;

//}

game->scrollx=-game->fbird.x+300;

return done;}

int main(int argc, char \*argv[])

{

SDL\_Window \*window=NULL; //declare a window

SDL\_Renderer \*renderer=NULL; //declare a renderer

SDL\_Init(SDL\_INIT\_VIDEO |SDL\_INIT\_AUDIO); //initialize SDL2

//create an application window with the following settings:

window =SDL\_CreateWindow("Flappy Bird",SDL\_WINDOWPOS\_CENTERED,SDL\_WINDOWPOS\_CENTERED,640,480,0);

renderer =SDL\_CreateRenderer(window,-1,SDL\_RENDERER\_ACCELERATED| SDL\_RENDERER\_PRESENTVSYNC);

GameState game;

game.renderer =renderer;

game.scrollx = 0;

game.fbird.jump=true;

game.fbird.animframe=0;

game.time =0;

game.font =NULL;

game.bgMusic =NULL;

Mix\_OpenAudio(MIX\_DEFAULT\_FREQUENCY,MIX\_DEFAULT\_FORMAT,MIX\_DEFAULT\_CHANNELS,4096);//initialize audio

TTF\_Init();

loadGame(&game);

int done =0;

Mix\_PlayMusic(game.bgMusic,-1);

while(!done){

//check for events

done= processEvents(window,&game);

process(&game);

//detects collision

collisionDetect(&game);

//render display

doRender(renderer,&game);

//delay

// SDL\_Delay(10);

}

//initialize ttf font

//shutdown game and unload all memory

SDL\_DestroyTexture(game.wall);

SDL\_DestroyTexture(game.ground);

SDL\_DestroyTexture(game.bird1);

SDL\_DestroyTexture(game.bird2);

if (game.label!=NULL)

SDL\_DestroyTexture(game.font);

TTF\_CloseFont(game.font);

//close and destroy the windows

Mix\_FreeMusic(game.bgMusic);

SDL\_DestroyWindow(window);

SDL\_DestroyRenderer(renderer);

Mix\_CloseAudio();

TTF\_Quit();

//clean up

SDL\_Quit();

return 0;

}

**For the header #include “main.h”**:

//main.h

#ifndef sdl2\_testing\_main\_h

#define sdl2\_testing\_main\_h

#include<SDL2/SDL.h>

#include<SDL2/SDL\_image.h>

#include<SDL2/SDL\_mixer.h>

#include<stdbool.h>

typedef struct

{

float x, y ,dy;

bool jump;

short life;

int animframe;

char \*name;

}Bird;

typedef struct

{

int x,y1,y2,w,h1,h2;

}Wall;

typedef struct

{

float scrollx;

Wall walls[500];

//the bird

Bird fbird;

//renderer

SDL\_Renderer \*renderer;

//time for animations

int time;

//images

SDL\_Texture \*ground;

SDL\_Texture \*bird1;

SDL\_Texture \*bird2;

SDL\_Texture \*wall;

SDL\_Texture \*label;

//sounds

int musicChannel;

Mix\_Music \*bgMusic;

Mix\_Chunk \*jumpSound, \*wallColi;

//Fonts

TTF\_Font \*font;

}GameState;

#endif // sdl2\_testing\_main\_h