



SUBMITTED TO:

Ma’am Sanya Zafar

SUBMITTED BY:

Furqan Ali (200901073)

Vanya Ahmed (200901095)

Haseeb-ul-Hassan (200901029)

Namrah Waseem (200901035)

Zooia Zainab (200901034)

SECTION:

BS-CS-01-B

COURSE:

Programming Fundamentals Lab

SUBMISSION DATE:  
29th December 2020

BS Computer Science

PROJECT REPORT

IN DEPTH DETAIL:

INTRODUCTION:

Our project is a 2D game in which our hero/champion is defending his hometown from the Demon Lord’s minions who are attacking and destroying his hometown. The Hero is fighting against the minions who are rampaging his hometown. Whenever the hero kills one of the minions of the demon lord, he is awarded 10 points.

TECHNIQUES USED:

In our project we used Visual Studio 2019 as a platform to write the code of our project.

In our project we used basic commands of C++ which are cout, cin, while loop, for loop etc.

Our focus was on the SFML (Simple and Fast Multimedia Library) which is the key to success in completing

our project. We used the Graphic & Window library from SFML.

We also used the knowledge of Classes in our project and made three Classes named Animation, Enemy

& Hero.

Project:

Our project is divided into four Classes which are Game1, Animation, Enemy & Hero.

We will be discussing each Class.

The main functions:

|  |
| --- |
| CODE:  *#include<SFML/Graphics.hpp>*  *#include<SFML/Window.hpp>*  *#include<SFML/System.hpp>*  *#include"Enemy.h"*  *#include<ctime>*  *#include"Hero.h"*  *#include<iostream>*  *#include<string>*  *using namespace sf;*  *using namespace std;*  *int main()*  *{*  *std::string y;*  *cout << "...................................\"How to play the game\"...................................\n";*  *cout << ".....................................\"For movemennt\"........................................\n";*  *cout << ".....................................\"W for going Up\".......................................\n";*  *cout << ".....................................\"S for going Down\".....................................\n";*  *cout << ".....................................\"A for going Left\".....................................\n";*  *cout << ".....................................\"D for going Right\"....................................\n";*  *cout << "...............................\"hold J for attacking the enemy\".............................\n";*  *cout << "...............................\"For each kill you get 10 points\"............................\n";*  *cout << "....................\"If 10 enemies passes you and get to the castle you lose\"................\n";*  *cout << "..............................\"Hopefully you will enjoy the game\"...........................\n";*  *cout << "\n\nPlease enter \"Your name\" to start the game: ";*  *cin >> y;*      *RenderWindow window(VideoMode(800, 600), "THE RISE OF THE HERO", Style::Close|Style::Fullscreen);*  *window.setFramerateLimit(60);*  *//enemy*  *Texture text;*  *text.loadFromFile("content/namra.png");*  *Enemy enemy(&text, Vector2u(4, 1), 0.15);*    *//hero*  *Texture vani;*  *vani.loadFromFile("content/vani.png");*  *Hero hero(&vani, Vector2u(5, 5), 0.5);*      *//background*  *Texture bg;*  *bg.loadFromFile("content/bg01.png");*  *RectangleShape bx;*  *bx.setTexture(&bg);*  *bx.setSize(Vector2f(800, 600));*    *float deltatime;*  *Clock cll;*    *while (window.isOpen())*  *{*  *if (enemy.end() <= 10)*  *{*  *deltatime = cll.restart().asSeconds();*  *enemy.update(deltatime, &window); hero.updatehero(deltatime, &window);*    *///clearing*  *window.clear();*  *//draw*  *window.draw(bx);*  *enemy.render(&window);*  *hero.renderhero(&window);*  *//displaying*  *window.display();*  *}*  *else*  *{*  *window.close();*  *}*    *}*  *cout << y << " your score is " << enemy.getpoint()<<endl;*  *}* |

Explanation:

In the Main function we are using SFML Graphics, Window& System library we have first shown the rules and the buttons that are going to be used in the future to play the game. After showing the rules we have rendered or drawn our window that is going to hold all the animations in it. After our window has been made, we have run a while loop which is going to continuously run until the window has been closed.

In our main function we are calling all the other classes and using their function to perfume specific takes like updating the characters and drawing the characters on the window.

Our main function is also used to load the background image and also the score that our hero gets after killing an enemy.

Enemy Class:

Enemy.h:

|  |
| --- |
| CODE:  *#pragma once*  *#include<SFML/Graphics.hpp>*  *#include<SFML/Window.hpp>*  *#include<SFML/System.hpp>*  *#include<iostream>*  *#include"Animation.h"*  *#include"Hero.h"*  *using namespace std;*  *using namespace sf;*  *class Enemy*  *{*  *public:*  *//publlic funtions*  *//constructor/destructors*  *Enemy(Texture\* text, Vector2u imagecount, float switchtime);*  *~Enemy();*  *//functions......*  *void Enemyspawn(RenderWindow\* window);*  *//updating enemy*  *void updateEnemy(float deltatime, RenderWindow\* window);*  *//Rendering enemy*  *void renderEnemy(RenderWindow\* window);*  *//updating*  *void update(float deltatime, RenderWindow\* window);*  *//rendering*  *void render(RenderWindow \*window);*  *//doing anything*  *void pollevent( RenderWindow\* window);*  *//mouseposition*  *void mousepositon(RenderWindow\* window);*  *//geting points*  *float getpoint();*  *//ending*  *int end();*  *//private functions*  *private:*  *void intivariable();*  *void intienemy();*    *//public vaiable*  *public:*  *//private variable*  *private:*  *Event ev;*  *//Game logic*  *int point;*  *float enemyspawntimer;*  *float enemyspawntimer\_max;*  *int maxenemies;*  *int loss=0;*  *//mous position*  *Vector2i mouseposwindow;*    *Vector2f mouseposview;*  *// Game objects*  *std::vector <RectangleShape>enemies;*  *RectangleShape enemy;*  *RectangleShape hero;*  *//other class*  *Animation animation;*  *Hero herro;*  *};* |

Explanation:

In the header file of the Enemy class, we are defining some variables & functions that are going to be used in the .cpp file later.

Enemy.cpp:

|  |
| --- |
| CODE:  *#include "Enemy.h"*  *#include <ctime>*  *#include <thread>*  *#include<chrono>*  *Enemy::Enemy(Texture\* text, Vector2u imagecount, float switchtime):*  *animation(text,imagecount,switchtime),herro(text,imagecount,switchtime)*  *{*    *this->intivariable();*  *this->intienemy();*  *this->mouseposwindow;*  *this->enemy.setOrigin(enemy.getSize() / 2.0f);*  *this->enemy.setTexture(text);*  *}*  *Enemy::~Enemy()*  *{*  *}*  *void Enemy::intivariable()*  *{*    *int point();*  *float enemyspawntimer();*  *float enemyspawntimer\_max();*  *int maxenemies();*  *//Game logic*  *this->point = 0;*    *this->enemyspawntimer\_max = 40.f;*  *this->enemyspawntimer = this->enemyspawntimer\_max;*  *this->maxenemies = 10;*  *}*  *void Enemy::intienemy()*  *{*  *Texture text;*  *enemy.setPosition(0.0f, 0.0f);*  *this->enemy.setSize(Vector2f(90.f, 260.f));*  *}*  *//window*  *//rendering window*  *void Enemy::render(RenderWindow\* window)*  *{*  *//drawing*  *this->renderEnemy(window);*  *}*  *//updating window*  *void Enemy::update(float deltatime, RenderWindow\* window)*  *{*  *//calling the events*  *this->pollevent(window);*  *//mousepostion*  *this->mousepositon( window);*  *//updating the eneming*  *this->updateEnemy( deltatime,window);*  *}*  *//doing something / events*  *void Enemy::pollevent(RenderWindow \*window)*  *{*    *//taking some action / event*  *while (window->pollEvent(this->ev))*  *{*  *switch (ev.type)*  *{*  *case Event::Closed:*  *{*  *window->close();*  *}*  *case Event::KeyPressed:*  *{*    *if (ev.key.code == Keyboard::Escape)*  *{*  *window->close();*  *break;*  *}*  *}*  *default:*  *break;*  *}*  *}*  *}*  *//mouse postion*  *void Enemy::mousepositon(RenderWindow \*window)*  *{*  *this->mouseposwindow = Mouse::getPosition(\*window);*  *this->mouseposview = window->mapPixelToCoords(this->mouseposwindow);*  *}*  *//Enemies*  *void Enemy::Enemyspawn(RenderWindow\* window)*  *{*  *this->enemy.setPosition(*  *800,*  *20+(static\_cast<float>(rand() % static\_cast<int>(window->getSize().y - this->enemy.getSize().y))*  *));*    *//span the enemy*  *this->enemies.push\_back(this->enemy);*  *}*  *//updating enemies*  *void Enemy::updateEnemy(float deltatime, RenderWindow\* window)*  *{ animation.update(0, deltatime, true);*  *enemy.setTextureRect(animation.uvRect);*  *//updating the timer for enemy spawning*  *if (this->enemies.size() < this->maxenemies)*  *{*  *if (this->enemyspawntimer >= this->enemyspawntimer\_max)*  *{*  *//spawning the enemies*  *this->Enemyspawn(window);*  *this->enemyspawntimer = 0.f;*  *}*  *else*  *{*  *this->enemyspawntimer += 1.f;*  *}*  *}*  *//moves and udates the enemies*  *for (int i = 0; i < this->enemies.size(); i++)*  *{*  *bool deleted = false;*  *this->enemies[i].move(-3.5f, 0.f);*  *this->enemies[i].setTextureRect(animation.uvRect);*  *//check if mouse is click on the enemy*  *Vector2f getpos;*  *getpos = herro.Getpos();*  *Vector2f poss;*  *poss = enemies[i].getPosition();*  *if (Keyboard::isKeyPressed(Keyboard::J))*  *{*  *if (this->enemies[i].getGlobalBounds().contains(this->mouseposview.x,(this->mouseposview.y-120)))*  *{*  *deleted = true;*  *//Gain points*  *this->point += 10;*  *}*  *}*  *//if the enemy pass the left side of the screen*  *if ((enemies[i].getPosition().x<-1))*  *{*  *this->enemies.erase(this->enemies.begin() + i);*  *this->loss++;*  *}*  *//erase the enemy*  *if (deleted)*  *{*  *this->enemies.erase(this->enemies.begin() + i);*  *}*      *}*    *}//updateing enemy end*  *void Enemy::renderEnemy(RenderWindow\* window)*  *{*  *//drawing all the enemies*  *for (auto& e : this->enemies)*  *{*  *window->draw(e);*  *}*  *}*  *//storing points*  *float Enemy::getpoint()*  *{*  *return this->point;*  *}*  *//ending condition*  *int Enemy::end()*  *{*  *return this->loss;*  *}* |

Explanation:

In the .cpp file we are using the functions that were declared in the header file. We have made different functions like update function which is being used to update the enemy/move the enemy and poll event.

which is responsible for noting down and respond accordingly to any event that take place like Closing the window.

Enemyspawn function is being used to spawn enemy on the window with position which is fixed on the x-axis but random on the y-axis.

updateEnemy function is being used to move the enemy on the x-axis automatically without any key pressed with a fix speed and also delete enemy when the user press J key in a specific area.

renderEnemy function is being used to draw the enemy on the window.

Some other functions are also being used to get the location of mouse, to get the points and to count the enemies which have crossed a specific point which, is also used to end the game when 10 enemies passed that point.

Other functions like intivariable and intienemy are being used to define some variables and the shape of the enemy.

Animation Class:

Animation.h:

|  |
| --- |
| CODE:  *#pragma once*  *#include<SFML/Graphics.hpp>*  *using namespace sf;*  *class Animation*  *{*  *//functions*  *public:*  *//counstructor/destructor*  *Animation(Texture\* text, Vector2u imagecount, float swtichtime);*  *~Animation();*  *//funtions*  *void update(int row, float deltatime,bool faceright);*  *private:*  *//variables*  *public:*  *IntRect uvRect;*  *private:*  *Vector2u imagecount;*  *Vector2u currentimage;*  *float switchtime;*  *double time;*  *};* |

Explanation:

In the header file of the Animation class we are declaring some variables and single functions.

Animation.cpp:

|  |
| --- |
| CODE:  *#include "Animation.h"*  *Animation::Animation(Texture\* text, Vector2u imagecount,float swtichtime)*  *{*  *this->switchtime = swtichtime;*  *this->imagecount = imagecount;*  *time = 0.0f;*  *currentimage.x = 0;*    *uvRect.width =static\_cast<int>(text->getSize().x /float(imagecount.x));*  *uvRect.height = static\_cast<int>( text->getSize().y /float(imagecount.y));*  *}*  *Animation::~Animation()*  *{*  *}*  *void Animation::update(int row, float deltatime, bool faceright)*  *{*    *currentimage.y = row;*  *time +=deltatime;*    *if (time > switchtime)*  *{*  *time -=switchtime ;*  *currentimage.x++;*  *if (currentimage.x>=imagecount.x)*  *{*  *currentimage.x = 0;*  *}*  *}*  *uvRect.top = currentimage.y \* uvRect.height;*  *if (faceright)*  *{*  *uvRect.left = currentimage.x \* uvRect.width;*  *uvRect.width = abs(uvRect.width);*  *}*  *else*  *{*  *uvRect.left = (currentimage.x + 1) \* abs(uvRect.width);*  *uvRect.width = -abs(uvRect.width);*  *}*    *}* |

Explanation:

In Animation.cpp we are defining values to variables in the constructor function which are going to help in the process of running the animation. The update function is basically changing the images from the sprite sheet which was load in with the Texture command.

Note:

Animation class is being used in Enemy and Hero class to load the image that we see on the screen and also the moving image that we see while moving or attacking the hero.

Hero Class:

Hero.h:

|  |
| --- |
| CODE:  *#pragma once*  *#include<SFML/Graphics.hpp>*  *#include "Animation.h"*  *#include<iostream>*  *using namespace sf;*  *class Hero*  *{*  *public:*  *Hero(Texture\* text, Vector2u imagecount, float switchtime);*  *~Hero();*  *void updatehero(float deltatime, RenderWindow\* window);*  *void renderhero(RenderWindow\* window);*  *Vector2f Getpos() { return body.getPosition(); }*  *private:*  *RectangleShape body;*  *Animation animation;*  *};* |

Explanation:

In the header file of the Hero Class we are declaring some functions and variable and also calling the Animation Class.

Hero.cpp:

|  |
| --- |
| CODE:  *#include "Hero.h"*  *Hero::Hero(Texture\* text, Vector2u imagecount, float switchtime):*  *animation(text,imagecount,switchtime)*  *{*  *body.setPosition(200.0f, 200.0f);*  *body.setSize(Vector2f(90.0, 100.0f));*  *body.setOrigin(body.getSize() / 2.0f);*  *body.setTexture(text);*  *}*  *Hero::~Hero()*  *{*  *}*  *void Hero::updatehero(float deltatime,RenderWindow \*window)*  *{*  *bool faceright = true;*  *unsigned int row;*  *float speed = 100;*  *Vector2f movement(0.0f, 0.0f);*    *if (Keyboard::isKeyPressed(Keyboard::Key::A))*  *{*  *faceright = false;*  *row = 0;*  *movement.x -= speed \* deltatime;*  *}*  *if (Keyboard::isKeyPressed(Keyboard::Key::D))*  *{*  *faceright = true;*  *row = 0;*  *movement.x += speed \* deltatime;*  *}*    *if (Keyboard::isKeyPressed(Keyboard::Key::W))*  *{*  *faceright = true;*  *row = 1;*  *movement.y -= speed \* deltatime;*  *}*  *if (Keyboard::isKeyPressed(Keyboard::Key::S))*  *{*  *row = 2;*  *movement.y += speed \* deltatime;*  *}*  *if (movement == Vector2f(0.0f, 0.0f))*  *{*  *row = 3;*  *}*  *if (Keyboard::isKeyPressed(Keyboard::J))*  *{*  *row = 4;*  *animation.update(row, deltatime, faceright);*  *}*  *animation.update(row, deltatime, faceright);*  *body.setTextureRect(animation.uvRect);*  *body.move(movement);*  *body.getGlobalBounds().contains(window->mapPixelToCoords(Mouse::getPosition(\*window)));*  *Mouse::getPosition(\*window);Mouse::setPosition(Vector2i(body.getPosition()+Vector2f(4.0f,0.0f)));*  *}*  *void Hero::renderhero(RenderWindow \*window)*  *{*  *window->draw(body);*  *}* |

Explanation:

In the cpp file of the Hero Class we are making our hero move and attack the enemy. In the constructor function we are defining the body and the animation of the hero.

In the update function we are setting up the conditions which must be fulfilled to make our hero move and attack the enemies. like, for movement we are specifying which specific key will move our hero in the given specific direction and which key or the mouse button is going to be used for attacking the enemy.

In the render function we are just drawing the hero on the screen as he moves on the window.

Screenshot of the project:





