***CAR GAME***

*Using SFML in Visual Studio*

***Caution: This will not run unless you setup SFML in your Project***

***And you must include pictures and fonts in project folder which I have used in this project***

***Font:***



*9 files*

**Game.h**

#pragma once

#pragma once

#include <SFML/Graphics.hpp>

#include<stdio.h>

#include<stdlib.h>

#include<iostream>

using namespace sf;

using namespace std;

class Game

{

int height, width;

public:

Game();

void DisplayInfo();

int getheight();

int getwidth();

};

**Game.cpp**

#include "Game.h"

Game::Game()

{

height = 480;

width = 640;

}

void Game::DisplayInfo()

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\nINSTRUCTION ABOUT GAMES\n");

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\nTHIS CAR GAME CONSIST OF 2 LEVELS WIN THE LEVEL ONE TO GET ACCESS TO THE LEVEL 2 \nWHERE YOU CAN DRIVE CAR WITHOUT ANY BOUNDS\n");

printf("You can quit the game any time by pressing escape button\n");

printf("REACH THE GREEN MARK TO WIN THE LEVEL ONE\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("Press \"R\" key to restart the level if you get out\n");

printf("Function of different keys are as follow\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" Escape = exiting\n arrow keys = movement\n Restart level\t=\tRestart \n");

}

int Game::getheight()

{

return height;

}

int Game::getwidth()

{

return width;

}

**Texture.h**

#pragma once

#include"Game.h"

const int MAX = 3;

class Textures

{

public:

Texture background1, background2, gameover, win, texture;

Sprite spGameover, spwin, spBackground2, spritee;

void loadTextures();

int selectedItemIndex = 0;

Font font;

Text menu[MAX];

void setMainMenu(float width, float height);

void draw(RenderWindow& window);

void Moveup();

void MoveDown();

int getPressedItem();

};

**Texture,cpp**

#pragma warning(disable:4996)

#include "Textures.h"

void Textures::loadTextures()

{

if (!gameover.loadFromFile("Texture/gg.jpg"))

printf("Error loading gameover image");

spGameover.setTexture(gameover);

spGameover.setScale(Vector2f(0.6, 0.65));

if (!win.loadFromFile("Texture/win.jpg"))

printf("Error loading win image");

spwin.setTexture(win);

if (!background2.loadFromFile("Texture/background2.png"))

printf("Error loading bacjground image");

spBackground2.setTexture(background2);

if (!texture.loadFromFile("Texture/cargame.jpg"))

printf("Error loading gameover image");

spritee.setTexture(texture);

}

void Textures::setMainMenu(float width, float height)

{

if (!font.loadFromFile("Times-New-Romance.ttf"))

{

//ERROR DETECTION

}

menu[0].setFont(font);

menu[0].setColor(Color::Blue);

menu[0].setString("--PLAY--");

menu[0].setPosition(Vector2f((width - 15 \* strlen("--PLAY--")) / 2, height / (MAX + 1) \* 1));

menu[1].setFont(font);

menu[1].setColor(Color::White);

menu[1].setString("--HELP--");

menu[1].setPosition(Vector2f((width - 15 \* strlen("--ABOUT--")) / 2, height / (MAX + 1) \* 2));

menu[2].setFont(font);

menu[2].setColor(Color::White);

menu[2].setString("--EXIT--");

menu[2].setPosition(Vector2f((width - 15 \* strlen("--EXIT--")) / 2, height / (MAX + 1) \* 3));

}

void Textures::draw(RenderWindow& window)

{

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

{

window.draw(menu[i]);

}

}

void Textures::Moveup()

{

menu[selectedItemIndex].setColor(Color::White);

selectedItemIndex--;

menu[selectedItemIndex].setColor(Color::Blue);

}

void Textures::MoveDown()

{

if (selectedItemIndex + 1 < MAX)

{

menu[selectedItemIndex].setColor(Color::White);

selectedItemIndex++;

menu[selectedItemIndex].setColor(Color::Blue);

}

}

int Textures::getPressedItem()

{

return selectedItemIndex;

return 0;

}

**Car.h**

#pragma once

#include"Game.h"

#include"Textures.h"

class Car

{

public:

Texture car1;

Sprite spcar1;

void loadTex();

};

**Car.cpp**

#include "Car.h"

void Car::loadTex()

{

Textures tx;

car1.loadFromFile("Texture/car1.PNG");

spcar1.setTexture(car1);

spcar1.setScale(0.8, 0.8);

spcar1.setOrigin(Vector2f(spcar1.getTexture()->getSize().x \* 0.5, spcar1.getTexture()->getSize().y \* 0.5));

}

**Hurdles.h**

#pragma once

#include"Game.h"

#include"Textures.h"

#include"Car.h"

class Hurdles

{

public:

RectangleShape rect();

RectangleShape rect1();

RectangleShape rect2();

RectangleShape rect3();

RectangleShape rectangle1();

RectangleShape rectangle2();

RectangleShape rectangle3();

RectangleShape rectangle4();

RectangleShape rectangle5();

RectangleShape rectangle7();

RectangleShape rectangle8();

CircleShape circle1();

RectangleShape rectangle91();

RectangleShape rectangle10();

};

**Hurdles.cpp**

#include "Hurdles.h"

RectangleShape Hurdles::rect()

{

RectangleShape rect;

rect.setPosition(0, 0);

rect.setSize(Vector2f(20, 480));

rect.setFillColor(Color::Red);

return rect;

}

RectangleShape Hurdles::rect1()

{

RectangleShape rect1;

rect1.setPosition(0, 0);

rect1.setSize(Vector2f(640, 20));

rect1.setFillColor(Color::Red);

return rect1;

}

RectangleShape Hurdles::rect2()

{

RectangleShape rect2;

rect2.setPosition(620, 0);

rect2.setSize(Vector2f(20, 480));

rect2.setFillColor(Color::Red);

return rect2;

}

RectangleShape Hurdles::rect3()

{

RectangleShape rect3;

rect3.setPosition(0, 460);

rect3.setSize(Vector2f(640, 20));

rect3.setFillColor(Color::Red);

return rect3;

}

RectangleShape Hurdles::rectangle1()

{

RectangleShape rectangle1;

rectangle1.setPosition(70, 280);

rectangle1.setSize(Vector2f(20, 180));

rectangle1.setFillColor(Color::Red);

return rectangle1;

}

RectangleShape Hurdles::rectangle2()

{

RectangleShape rectangle2;

rectangle2.setPosition(190, 180);

rectangle2.setSize(Vector2f(20, 220));

rectangle2.setFillColor(Color::Yellow);

return rectangle2;

}

RectangleShape Hurdles::rectangle3()

{

RectangleShape rectangle3;

rectangle3.setPosition(210, 380);

rectangle3.setSize(Vector2f(310, 20));

rectangle3.setFillColor(Color::Red);

return rectangle3;

}

RectangleShape Hurdles::rectangle4()

{

RectangleShape rectangle4;

rectangle4.setPosition(300, 300);

rectangle4.setSize(Vector2f(320, 20));

rectangle4.setFillColor(Color::Yellow);

return rectangle4;

}

RectangleShape Hurdles::rectangle5()

{

RectangleShape rectangle5;

rectangle5.setPosition(210, 180);

rectangle5.setSize(Vector2f(200, 20));

rectangle5.setFillColor(Color::Yellow);

return rectangle5;

}

RectangleShape Hurdles::rectangle7()

{

RectangleShape rectangle7;

rectangle7.setPosition(410, 170);

rectangle7.setSize(Vector2f(20, 60));

rectangle7.setFillColor(Color::Red);

return rectangle7;

}

RectangleShape Hurdles::rectangle8()

{

RectangleShape rectangle8;

rectangle8.setPosition(20, 180);

rectangle8.setSize(Vector2f(170, 20));

rectangle8.setFillColor(Color::Yellow);

return rectangle8;

}

CircleShape Hurdles::circle1()

{

CircleShape circle1;

circle1.setPosition(490, 170);

circle1.setRadius(30);

circle1.setFillColor(Color::Red);

return circle1;

}

RectangleShape Hurdles::rectangle91()

{

RectangleShape rectangle91;

rectangle91.setPosition(100, 90);

rectangle91.setSize(Vector2f(520, 20));

rectangle91.setFillColor(Color::Red);

return rectangle91;

}

RectangleShape Hurdles::rectangle10()

{

RectangleShape rectangle10;

rectangle10.setPosition(500, 25);

rectangle10.setSize(Vector2f(100, 60));

rectangle10.setOutlineColor(Color::White);

rectangle10.setFillColor(Color::Green);

return rectangle10;

}

**Source.cpp**

#include"Game.h"

#include"Textures.h"

#include"Hurdles.h"

#include"Car.h"

int Gameover(RenderWindow& window, Event& e, Textures& T)

{

window.clear();

window.draw(T.spGameover);

while (window.pollEvent(e))

{

int count = 0;

int counter = 1;

if (counter == 1)

printf("Try again");

counter++;

if (e.key.code == Keyboard::Return)

{

count++;

break;

}

else if (e.key.code == Keyboard::Escape)

window.close();

return count;

}

}

int main()

{//object creations and calling default constructor

Game gObj;

Hurdles Hobj;

Car Cobj;

Textures Tobj;

RenderWindow window(VideoMode(gObj.getwidth(), gObj.getheight()), "Car Game");

window.setFramerateLimit(120);

Tobj.setMainMenu(window.getSize().x, window.getSize().y);

//for getting car and others

Cobj.loadTex();

Tobj.loadTextures();

//FOR MENU

Event E;

while (window.isOpen())

{

//polling

while (window.pollEvent(E))

{

switch (E.type)

{

case Event::KeyReleased:

switch (E.key.code)

{

case Keyboard::Up:

Tobj.Moveup(); break;

case Keyboard::Down:

Tobj.MoveDown(); break;

case Keyboard::Return:

switch (Tobj.getPressedItem())

{

case 0:

{

cout << "PLAY button Entered\n";

goto start;

break;

}

case 1:

{

cout << "ABOUT button Entered\n";

gObj.DisplayInfo();

break;

}

case 2:window.close();

}

break;

}

break;

case Event::Closed:

window.close(); break;

}

}

//update

//render

window.clear();//clear old frame

window.draw(Tobj.spritee);//draw picture

Tobj.draw(window);//draw text

window.display();//tell window is done drawing

}

system("pause");

//haseeb

start:

float a = 50, b = 410, x = 300, y = 300;

float speed = 0, angle = 0;

float maxSpeed = 3.0;

float acc = 0.01, dec = 0.02;

float turnSpeed = 0.08;

int offsetX = 0, offsetY = 0;

Event e;

//game loop

while (window.isOpen())

{

//for closing of window

Event evnt;

while (window.pollEvent(evnt))

{

if (evnt.type == Event::Closed)

window.close();

if (evnt.key.code == Keyboard::Escape)

window.close();

}

bool Up = 0, Right = 0, Down = 0, Left = 0;

if (Keyboard::isKeyPressed(Keyboard::Up)) Up = 1;

if (Keyboard::isKeyPressed(Keyboard::Right)) Right = 1;

if (Keyboard::isKeyPressed(Keyboard::Down)) Down = 1;

if (Keyboard::isKeyPressed(Keyboard::Left)) Left = 1;

if (Up && speed < maxSpeed)

if (speed < 0) speed += dec;

else speed += acc;

if (Down && speed > -maxSpeed)

if (speed > 0) speed -= dec;

else speed -= acc;

if (!Up && !Down)

if (speed - dec > 0) speed -= dec;

else if (speed + dec < 0) speed += dec;

else speed = 0;

if (Right && speed != 0) angle += turnSpeed \* speed / maxSpeed;

if (Left && speed != 0) angle -= turnSpeed \* speed / maxSpeed;

a += sin(angle) \* speed;

b -= cos(angle) \* speed;

//Draw Background

window.clear(Color::Black);

//drawing hurdles

window.draw(Hobj.rect());

window.draw(Hobj.rect1());

window.draw(Hobj.rect2());

window.draw(Hobj.rect3());

window.draw(Hobj.rectangle1());

window.draw(Hobj.rectangle2());

window.draw(Hobj.rectangle3());

window.draw(Hobj.rectangle4());

window.draw(Hobj.rectangle5());

window.draw(Hobj.rectangle7());

window.draw(Hobj.rectangle8());

window.draw(Hobj.circle1());

window.draw(Hobj.rectangle91());

window.draw(Hobj.rectangle10());

//COLLOSIONS

//left

if (Cobj.spcar1.getGlobalBounds().intersects(Hobj.rect().getGlobalBounds()))

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//Top collosion

if (Cobj.spcar1.getGlobalBounds().intersects(Hobj.rect1().getGlobalBounds()))

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//right collosion

if (Cobj.spcar1.getGlobalBounds().intersects(Hobj.rect2().getGlobalBounds()))

{

window.clear();

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//down collosion

if (Cobj.spcar1.getGlobalBounds().intersects(Hobj.rect3().getGlobalBounds()))

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//1st wall

if ((Cobj.spcar1.getPosition().x) <= 90.f && (Cobj.spcar1.getPosition().x) >= 70.f)

if ((Cobj.spcar1.getPosition().y) >= 280.f && (Cobj.spcar1.getPosition().y) <= 460.f)

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//2nd wall

if (Cobj.spcar1.getPosition().x <= 210.f && Cobj.spcar1.getPosition().x >= 180.f)

if (Cobj.spcar1.getPosition().y >= 280.f && Cobj.spcar1.getPosition().y <= 400.f)

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//3rd wall

if (Cobj.spcar1.getPosition().x <= 550.f && Cobj.spcar1.getPosition().x >= 210.f)

if (Cobj.spcar1.getPosition().y >= 380.f && Cobj.spcar1.getPosition().y <= 395.f)

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//4th wall

if (Cobj.spcar1.getPosition().x <= 620.f && Cobj.spcar1.getPosition().x >= 300.f)

if (Cobj.spcar1.getPosition().y >= 300.f && Cobj.spcar1.getPosition().y <= 320.f)

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//5th

//6th

if (Cobj.spcar1.getPosition().x <= 410.f && Cobj.spcar1.getPosition().x >= 210.f)

if (Cobj.spcar1.getPosition().y >= 170.f && Cobj.spcar1.getPosition().y <= 190.f)

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//7th

if (Cobj.spcar1.getPosition().x <= 430.f && Cobj.spcar1.getPosition().x >= 410.f)

if (Cobj.spcar1.getPosition().y >= 230.f && Cobj.spcar1.getPosition().y <= 170.f)

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//8th

if (Cobj.spcar1.getPosition().x <= 190.f && Cobj.spcar1.getPosition().x >= 20.f)

if (Cobj.spcar1.getPosition().y >= 180.f && Cobj.spcar1.getPosition().y <= 200.f)

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//9th

if (Cobj.spcar1.getPosition().x <= 620.f && Cobj.spcar1.getPosition().x >= 100.f)

if (Cobj.spcar1.getPosition().y >= 90.f && Cobj.spcar1.getPosition().y <= 110.f)

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//circle

if (Cobj.spcar1.getPosition().x <= 520.f && Cobj.spcar1.getPosition().x >= 490.f)

if (Cobj.spcar1.getPosition().y >= 170.f && Cobj.spcar1.getPosition().y <= 210.f)

{

int count = Gameover(window, e, Tobj);

if (count == 1)

goto start;

}

//For winning

if (Cobj.spcar1.getPosition().x > 515 && Cobj.spcar1.getPosition().x < 600)

if (Cobj.spcar1.getPosition().y > 25 && Cobj.spcar1.getPosition().y < 80)

{

window.clear(Color::Green);

window.draw(Tobj.spwin);

while (window.pollEvent(e))

{

if (e.key.code == Keyboard::Return|| e.key.code == Keyboard::R)

goto level2;

}

}

Cobj.spcar1.setPosition(a, b);

Cobj.spcar1.setRotation(angle \* 180 / 3.14);

Cobj.spcar1.setColor(Color::Green);

window.draw(Cobj.spcar1);

window.display();

}

level2:

//2nd level just for random driving

while (window.isOpen())

{

Event evnt;

while (window.pollEvent(evnt))

{

if (evnt.type == Event::Closed)

window.close();

if (evnt.key.code == Keyboard::Escape)

window.close();

}

//movement

bool Up = 0, Right = 0, Down = 0, Left = 0;

if (Keyboard::isKeyPressed(Keyboard::Up)) Up = 1;

if (Keyboard::isKeyPressed(Keyboard::Right)) Right = 1;

if (Keyboard::isKeyPressed(Keyboard::Down)) Down = 1;

if (Keyboard::isKeyPressed(Keyboard::Left)) Left = 1;

if (Up && speed < maxSpeed)

if (speed < 0) speed += dec;

else speed += acc;

if (Down && speed > -maxSpeed)

if (speed > 0) speed -= dec;

else speed -= acc;

if (!Up && !Down)

if (speed - dec > 0) speed -= dec;

else if (speed + dec < 0) speed += dec;

else speed = 0;

if (Right && speed != 0) angle += turnSpeed \* speed / maxSpeed;

if (Left && speed != 0) angle -= turnSpeed \* speed / maxSpeed;

x += sin(angle) \* speed;

y -= cos(angle) \* speed;

//for making map follow according to the car moving

if (x > 320) offsetX = x - 320;

if (y > 240) offsetY = y - 240;

//Draw Background

window.clear(Color::Red);

Tobj.spBackground2.setPosition(-offsetX, -offsetY);

window.draw(Tobj.spBackground2);

//Draw Car

Cobj.spcar1.setPosition(x - offsetX, y - offsetY);

Cobj.spcar1.setRotation(angle \* 180 / 3.14);

Cobj.spcar1.setColor(Color::Green);

window.draw(Cobj.spcar1);

window.display();

}//game loop ends

system("pause");

return 0;

}







