﻿using Microsoft.Xna.Framework;

using Microsoft.Xna.Framework.Content;

using Microsoft.Xna.Framework.Graphics;

using Microsoft.Xna.Framework.Input;

using MonoGame.Extended.Content; /\* LIBRARY was supposed to be used for Tmx file format from monogame extended.tiled \*/

using System.Collections.Generic;

using System;

using TiledSharp; /\* LIBRARY FOR EXPORTING TMX FILES IN MONOGAMES\*/

using Microsoft.Xna.Framework.Media;

namespace FL2

{

public class Game1 : Game

{

private GraphicsDeviceManager \_graphics;

private SpriteBatch \_spriteBatch;

/\*variables\*/

private Vector2 cameraPosition1; /\* varible for 2d camera \*/

private Player player1; /\*creating an object for the player \*/

List<coin> coins;

List<Enemy> enemies;

private Texture2D menuImage, endImage;

public enum GameState

{

Menu,

Playing,

End

}

private GameState \_currentState = GameState.Menu;

public KeyboardState \_previousKeyboardState;

/\* map \*/

TmxMap map; /\*variable for the map\*/

Texture2D tileset; /\*variable for tileset image \*/

int tileWidth; /\*variable for single tile width\*/

int tileHeight; /\*variable for single tile height\*/

int tilesetsize; /\*variable for total size of tile\*/

bool paused = false;

Texture2D pausedtexture;

Rectangle pausedrectangle;

Button btnPlay, btnQuit;

int screenHeight; /\* variable to calculate the screen height\*/

int screenWidth; /\*variable to caldulate the screen width \*/

private SpriteFont font;

public Game1()

{

\_graphics = new GraphicsDeviceManager(this);

Content.RootDirectory = "Content";

IsMouseVisible = true;

}

protected override void Initialize()

{

base.Initialize();

}

protected override void LoadContent()

{

\_spriteBatch = new SpriteBatch(GraphicsDevice);

screenWidth = GraphicsDevice.Viewport.Width;

screenHeight = GraphicsDevice.Viewport.Height;

/\*main menu\*/

menuImage = Content.Load<Texture2D>("menu");

endImage = Content.Load<Texture2D>("end");

map = new TmxMap("Content/map.tmx"); /\* loading TMX map \*/

tileset = Content.Load<Texture2D>(map.Tilesets[0].Name.ToString()); /\* picking tiles \*/

tileWidth = map.Tilesets[0].TileWidth; /\*calculating single tile width \*/

tileHeight = map.Tilesets[0].TileHeight; /\*calculating single tile height \*/

tilesetsize = tileset.Width / tileWidth; /\*calculating the total area of the tile \*/

Song song = Content.Load<Song>("music");

MediaPlayer.Play(song);

var playerTexture = Content.Load<Texture2D>("Emily"); /\* Adding texture to player \*/

player1 = new Player(playerTexture, new Vector2(200, 200)); /\* swpaning the player with texture and ingame location\*/

enemies = new List<Enemy>

{

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(600, 300), 250),

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(500, 550), 200),

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(600, 1050), 250),

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(600, 1750), 250),

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(600, 2700), 250),

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(900, 200), 250),

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(2200, 450), 250),

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(2500, 940), 250),

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(2300, 1390), 250),

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(2300, 1850), 250),

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(1800, 2950), 250),

new Enemy(Content.Load<Texture2D>("enemy3"), new Vector2(1200, 2800), 250),

};

var coinTexture = Content.Load<Texture2D>("coin");

coins = new List<coin>()

{

new coin(coinTexture, new Vector2(900, 800)),

new coin(coinTexture, new Vector2(500, 550)),

new coin(coinTexture, new Vector2(600, 1050)),

new coin(coinTexture, new Vector2(600, 1750)),

new coin(coinTexture, new Vector2(600, 2700)),

new coin(coinTexture, new Vector2(900, 200)),

new coin(coinTexture, new Vector2(2200, 450)),

new coin(coinTexture, new Vector2(2500, 940)),

new coin(coinTexture, new Vector2(2300, 1390)),

new coin(coinTexture, new Vector2(2300, 1850)),

new coin(coinTexture, new Vector2(2400, 2800)),

new coin(coinTexture, new Vector2(1800, 2950)),

new coin(coinTexture, new Vector2(1200, 2800)),

new coin(coinTexture, new Vector2(1200, 2490)),

new coin(coinTexture, new Vector2(1700, 2110)),

new coin(coinTexture, new Vector2(1300, 1850)),

new coin(coinTexture, new Vector2(1100, 1550)),

new coin(coinTexture, new Vector2(1300, 1000)),

new coin(coinTexture, new Vector2(700, 800)),

new coin(coinTexture, new Vector2(900, 800)),

};

IsMouseVisible = true;

pausedtexture = Content.Load<Texture2D>("pause");

pausedrectangle = new Rectangle(0, 0, pausedtexture.Width, pausedtexture.Height);

btnPlay = new Button();

btnPlay.Load(Content.Load<Texture2D>("play"), new Vector2(350, 225));

btnQuit = new Button();

btnQuit.Load(Content.Load<Texture2D>("stop"), new Vector2(350, 280));

}

protected override void Update(GameTime gameTime)

{

var keyboardState = Keyboard.GetState();

MouseState mouse = Mouse.GetState();

if (\_currentState == GameState.Menu)

{

if (keyboardState.IsKeyDown(Keys.Enter) && \_previousKeyboardState.IsKeyUp(Keys.Enter))

{

\_currentState = GameState.Playing;

}

}

else if (\_currentState == GameState.Playing)

{

if (!paused)

{

if (Keyboard.GetState().IsKeyDown(Keys.Escape))

{

paused = true;

btnPlay.isclicked = false;

}

player1.Update(enemies);

cameraPosition1 = new Vector2(player1.position.X - \_graphics.PreferredBackBufferWidth / 2,

player1.position.Y - \_graphics.PreferredBackBufferHeight / 2);

float maxCameraX = tileWidth \* map.Width - \_graphics.PreferredBackBufferWidth;

float maxCameraY = tileHeight \* map.Height - \_graphics.PreferredBackBufferHeight;

cameraPosition1 = new Vector2(MathHelper.Clamp(cameraPosition1.X, 0, maxCameraX),

MathHelper.Clamp(cameraPosition1.Y, 0, maxCameraY));

/\* update Enemies here \*/

foreach (var enemy in enemies)

{

enemy.Update();

}

if (player1.healthBar.Width < 10)

{

\_currentState = GameState.End;

}

foreach (coin coin in coins)

{

if (!coin.IsCollected && player1.BoundingRectangle.Intersects(coin.BoundingRectangle))

{

coin.IsCollected = true;

}

}

}

else if (paused)

{

if (btnPlay.isclicked)

paused = false;

if (btnQuit.isclicked)

Exit();

btnPlay.Update(mouse);

btnQuit.Update(mouse);

}

}

else if (\_currentState == GameState.End)

{

if (keyboardState.IsKeyDown(Keys.Enter) && \_previousKeyboardState.IsKeyUp(Keys.Enter))

{

\_currentState = GameState.Menu;

Exit();

}

}

\_previousKeyboardState = keyboardState;

base.Update(gameTime);

}

protected override void Draw(GameTime gameTime)

{

GraphicsDevice.Clear(Color.CornflowerBlue);

\_spriteBatch.Begin(SpriteSortMode.Deferred, BlendState.AlphaBlend, SamplerState.PointClamp, null, null, null, Matrix.CreateTranslation(-cameraPosition1.X, -cameraPosition1.Y, 0));

if (\_currentState == GameState.Menu)

{

\_spriteBatch.Draw(menuImage, Vector2.Zero, Color.White);

}

else if (\_currentState == GameState.Playing)

{

for (var j = 0; j < map.Layers.Count; j++)

{

for (var i = 0; i < map.Layers[j].Tiles.Count; i++)

{

int gid = map.Layers[j].Tiles[i].Gid;

if (gid == 0)

{

}

else

{

int tileFrame = gid - 1;

int column = tileFrame % tilesetsize;

int row = (int)Math.Floor((double)tileFrame / (double)tilesetsize);

float x = (i % map.Width) \* map.TileWidth;

float y = (float)Math.Floor(i / (double)map.Width) \* map.TileHeight;

Rectangle tilesetRec = new Rectangle(tileWidth \* column, tileHeight \* row, tileWidth, tileHeight);

\_spriteBatch.Draw(tileset, new Rectangle((int)x, (int)y, tileWidth, tileHeight), tilesetRec, Color.White);

}

}

}

player1.Draw(\_spriteBatch);

foreach (coin coin in coins)

{

if (!coin.IsCollected)

coin.Draw(\_spriteBatch);

}

foreach (var enemy in enemies)

{

enemy.Draw(\_spriteBatch);

}

if (paused)

{

\_spriteBatch.Draw(pausedtexture, new Rectangle((int)cameraPosition1.X, (int)cameraPosition1.Y, pausedrectangle.Width, pausedrectangle.Height), Color.White);

btnPlay.Draw(\_spriteBatch, cameraPosition1);

btnQuit.Draw(\_spriteBatch, cameraPosition1);

}

}

else if (\_currentState == GameState.End)

{

\_spriteBatch.Draw(endImage, new Rectangle((int)cameraPosition1.X, (int)cameraPosition1.Y, pausedrectangle.Width, pausedrectangle.Height), Color.White);

}

\_spriteBatch.End();

base.Draw(gameTime);

}

}

}