import flash.text.TextField;

import flash.utils.Timer;

import flash.events.TimerEvent;

import flash.events.KeyboardEvent;

import flash.events.Event;

import flash.ui.Keyboard;

import flash.display.MovieClip;

import flash.display3D.IndexBuffer3D;

import com.adobe.tvsdk.mediacore.utils.TimeRange;

import flash.media.Sound;

import flash.media.SoundChannel;

stop();

//Creating Variables

//constants

const LIVES: int = 5;

const INITIAL\_PLAYER\_XSPEED: int = 15;

const INITIAL\_PLAYER\_YSPEED: int = 15;

const PLAYER\_BULLETSPEED: int = 20;

const ENEMY\_TIMER: int = 1000;

const EXPLOSION\_TIMER: int = 350;

const HIT\_TIMER: int = 150;

const POWER\_TIMER: int = 10000;

const POWER\_DURATION\_TIMER: int = 3000;

const INITIAL\_SHOOTING\_SPEED: int = 125;

const ENEMY\_BASESPEED: int = 10;

const LEVEL\_1: int = 25;

const LEVEL\_2: int = 50;

const LEVEL\_3: int = 100;

const LEVEL\_BOSS: int = 500;

const BOUNDARY: int = 50;

//movieclips

var player: MovieClip;

var playerHit: MovieClip;

//textfields

var livesTxt: TextField;

var scoreTxt: TextField;

//booleans

var upPressed: Boolean;

var downPressed: Boolean;

var leftPressed: Boolean;

var rightPressed: Boolean;

var wPressed: Boolean;

var aPressed: Boolean;

var sPressed: Boolean;

var dPressed: Boolean;

var isShooting: Boolean;

//arrays

var playerBullets: Array = new Array();

var enemies: Array = new Array();

var explosions: Array = new Array();

var powers: Array = new Array();

//timers

var shootingTimer: Timer;

var enemyTimer: Timer;

var explosionTimer: Timer;

var playerHitTimer: Timer;

var spawnPowerUpTimer: Timer;

var powerUpTimer: Timer;

//numbers

var score: int;

var lives: int;

var shootingSpeed: int;

var playerXSpeed: int;

var playerYSpeed: int;

var leftSpawns: int;

var rightSpawns: int;

var upSpawns: int;

var downSpawns: int;

//sounds

var explosionSound: Sound = new ExplosionSound();

var playerHitSound: Sound = new PlayerHit1Sound();

var playerHitShieldSound: Sound = new PlayerHitShieldSound();

var playerShootSound: Sound = new PlayerShootSound();

var powerDownSound: Sound = new PowerDownSound();

var powerUpSound: Sound = new PowerUpSound();

var lifeUpSound: Sound = new LifeUpSound();

var powerFlybySound: Sound = new PowerFlybySound();

var gameOverSound: Sound = new GameOverSound();

var effectsChannel: SoundChannel;

//Initialise Game

initPlay();

function initPlay(): void

{

//setting intial values

stage.focus = stage;

upPressed = false;

downPressed = false;

leftPressed = false;

rightPressed = false;

wPressed = false;

aPressed = false;

sPressed = false;

dPressed = false;

isShooting = false;

playerHit.alpha = 0.5;

playerHit.visible = false;

playerBullets = [];

enemies = [];

explosions = [];

powers = [];

enemyTimer = new Timer(ENEMY\_TIMER); //ENEMY\_TIMER

enemyTimer.reset();

enemyTimer.start();

enemyTimer.addEventListener(TimerEvent.TIMER, spawnEnemy);

shootingTimer = new Timer(INITIAL\_SHOOTING\_SPEED);

shootingTimer.reset();

shootingTimer.start();

shootingTimer.addEventListener(TimerEvent.TIMER, checkShoot);

explosionTimer = new Timer(EXPLOSION\_TIMER);

explosionTimer.reset();

explosionTimer.addEventListener(TimerEvent.TIMER, removeExplosion);

playerHitTimer = new Timer(HIT\_TIMER);

playerHitTimer.reset();

playerHitTimer.addEventListener(TimerEvent.TIMER, removePlayerHit);

spawnPowerUpTimer = new Timer(POWER\_TIMER); //5000

spawnPowerUpTimer.reset();

spawnPowerUpTimer.start();

spawnPowerUpTimer.addEventListener(TimerEvent.TIMER, spawnPowers);

powerUpTimer = new Timer(POWER\_DURATION\_TIMER);

powerUpTimer.reset();

stage.addEventListener(KeyboardEvent.KEY\_DOWN, aKeyPressed);

stage.addEventListener(KeyboardEvent.KEY\_UP, aKeyReleased);

addEventListener(Event.ENTER\_FRAME, animate);

player.gotoAndStop(1);

player.x = stage.stageWidth / 2;

player.y = stage.stageHeight / 2;

lives = LIVES;

score = 0;

livesTxt.text = String(lives);

scoreTxt.text = String(score);

playerXSpeed = INITIAL\_PLAYER\_XSPEED;

playerYSpeed = INITIAL\_PLAYER\_YSPEED;

}

function animate(e: Event): void

{

movePlayer();

rotatePlayer();

moveBullet();

moveEnemy();

movePower();

bulletEnemyHitTest();

playerEnemyHitTest();

playerPowerHitTest();

update();

}

function movePower(): void

{

//loop through each power in powers array

for (var i: uint = 0; i < powers.length; i++)

{

//getting angle from current position to destination position

var deg = getAngle(powers[i].x, powers[i].y, powers[i].xPos, powers[i].yPos);

var rad: Number = degToRad(powers[i].rotation)

powers[i].rotation = deg;

//moving from current position to destination position

powers[i].x += Math.cos(rad) \* 10;

powers[i].y += Math.sin(rad) \* 10;

//removing power if it is off the stage

if (powers[i].y > stage.stageHeight + BOUNDARY)

{

removeChild(powers[i]);

powers.splice(i, 1);

}

else if (powers[i].y < 0 - BOUNDARY)

{

removeChild(powers[i]);

powers.splice(i, 1);

}

else if (powers[i].x < 0 - BOUNDARY)

{

removeChild(powers[i]);

powers.splice(i, 1);

}

else if (powers[i].x > stage.stageWidth + BOUNDARY)

{

removeChild(powers[i]);

powers.splice(i, 1);

}

}

}

function removePowers(e: TimerEvent): void

{

//reset all values on player that powers modify

player.shield = false;

shootingSpeed = INITIAL\_SHOOTING\_SPEED

shootingTimer.stop();

shootingTimer = new Timer(INITIAL\_SHOOTING\_SPEED);

shootingTimer.reset();

shootingTimer.start();

shootingTimer.addEventListener(TimerEvent.TIMER, checkShoot);

playerXSpeed = INITIAL\_PLAYER\_XSPEED;

playerYSpeed = INITIAL\_PLAYER\_YSPEED;

player.visible = true;

player.scaleX = .5;

player.scaleY = .5;

player.gotoAndStop(1);

effectsChannel.stop();

//resetting the power duration timer

powerUpTimer.addEventListener(TimerEvent.TIMER, removePowers);

powerUpTimer.reset();

}

//all the Power functions

function shield(): void

{

//turning shield on

player.shield = true;

player.gotoAndStop(2);

}

function shootSpeedUp(): void

{

//reducing the shooting timer to 0 to enable rapid fire

shootingSpeed = 0;

shootingTimer.stop();

shootingTimer = new Timer(shootingSpeed);

shootingTimer.reset();

shootingTimer.start();

shootingTimer.addEventListener(TimerEvent.TIMER, checkShoot);

}

function moveSpeedUp(): void

{

//increase the movement speed of the player

playerXSpeed = 25;

playerYSpeed = 25;

}

function addLife(): void

{

//add 1 life to current lives

lives++;

}

function shootSpeedDown(): void

{

//increasing the shooting timer to 500 to enable slow fire rate

shootingSpeed = 500;

shootingTimer.stop();

shootingTimer = new Timer(shootingSpeed);

shootingTimer.reset();

shootingTimer.start();

shootingTimer.addEventListener(TimerEvent.TIMER, checkShoot);

}

function moveSpeedDown(): void

{

//reduce the movement speed of the player

playerXSpeed = 10;

playerYSpeed = 10;

}

function invisiblePlayer(): void

{

//turn the player invisible

player.visible = false;

}

function largePlayer(): void

{

//enlarge the player

player.scaleX = 1.25;

player.scaleY = 1.25;

}

function playerPowerHitTest(): void

{

for (var i: int = 0; i < powers.length; i++)

{

//checking if player hits a power

if (player.hitZone.hitTestObject(powers[i]))

{

//checking which power to enable on the player

if (powers[i].power == "shield")

{

shield();

effectsChannel = powerUpSound.play(0, 2);

}

else if (powers[i].power == "shootUp")

{

shootSpeedUp();

effectsChannel = powerUpSound.play(0, 2);

}

else if (powers[i].power == "speedUp")

{

moveSpeedUp();

effectsChannel = powerUpSound.play(0, 2);

}

else if (powers[i].power == "addLife")

{

addLife();

effectsChannel = lifeUpSound.play();

}

else if (powers[i].power == "shootDown")

{

shootSpeedDown();

effectsChannel = powerDownSound.play(0, 2);

}

else if (powers[i].power == "speedDown")

{

moveSpeedDown();

effectsChannel = powerDownSound.play(0, 2);

}

else if (powers[i].power == "invisiblePlayer")

{

invisiblePlayer();

effectsChannel = powerDownSound.play(0, 2);

}

else if (powers[i].power == "largePlayer")

{

largePlayer();

effectsChannel = powerDownSound.play(0, 2);

}

//starting duration timer

powerUpTimer.start();

powerUpTimer.addEventListener(TimerEvent.TIMER, removePowers);

//remove power from stage

removeChild(powers[i]);

powers.splice(i, 1);

}

}

}

function spawnPowers(e: TimerEvent): void

{

//creating and spawning power on to stage and adding to powers array

var power: MovieClip = new Powers();

addChildAt(power, numChildren - 1);

powers.push(power);

//selecting random destination position

power.xPos = Math.random() \* stage.stageWidth;

power.yPos = Math.random() \* stage.stageHeight;

effectsChannel = powerFlybySound.play();

//Picking Spawn Location

var spawnLocation: Number = Math.random()

if (spawnLocation <= 0.25)

{

power.x = stage.stageWidth \* Math.random();

power.y = 0;

power.xPos = (stage.stageWidth \* Math.random()) + BOUNDARY;

power.yPos = stage.stageHeight + BOUNDARY;

}

else if (spawnLocation <= 0.50)

{

power.x = stage.stageWidth \* Math.random();

power.y = stage.stageHeight;

power.xPos = (stage.stageWidth \* Math.random()) + BOUNDARY;

power.yPos = 0 - BOUNDARY;

}

else if (spawnLocation <= 0.75)

{

power.x = 0;

power.y = stage.stageHeight \* Math.random();

power.xPos = stage.stageWidth + BOUNDARY;

power.yPos = (stage.stageHeight \* Math.random()) + BOUNDARY;

}

else

{

power.x = stage.stageWidth;

power.y = stage.stageHeight \* Math.random();

power.xPos = 0 - BOUNDARY;

power.yPos = (stage.stageHeight \* Math.random()) + BOUNDARY;

}

//Picking Power

var pickPosNeg: int = Math.round((Math.random()));

var pickPower: int;

if (pickPosNeg)

{

pickPower = Math.ceil(Math.random() \* 4);

if (pickPower == 1)

{

power.gotoAndStop("shootUp");

power.power = "shootUp";

}

else if (pickPower == 2)

{

power.gotoAndStop("speedUp");

power.power = "speedUp";

}

else if (pickPower == 3)

{

power.gotoAndStop("shield");

power.power = "shield";

}

else

{

power.gotoAndStop("addLife");

power.power = "addLife";

}

}

else

{

pickPower = Math.ceil(Math.random() \* 4);

if (pickPower == 1)

{

power.gotoAndStop("shootDown");

power.power = "shootDown";

}

else if (pickPower == 2)

{

power.gotoAndStop("speedDown");

power.power = "speedDown";

}

else if (pickPower == 3)

{

power.gotoAndStop("invisiblePlayer");

power.power = "invisiblePlayer";

}

else

{

power.gotoAndStop("largePlayer");

power.power = "largePlayer";

}

}

}

function gameOver(): void

{

//removing all objects from stage

var i: uint;

while ((enemies.length) || (explosions.length) || (playerBullets.length) || (powers.length))

{

for (i = 0; i < enemies.length; i++)

{

removeChild(enemies[i]);

enemies.splice(i, 1);

}

for (i = 0; i < explosions.length; i++)

{

removeChild(explosions[i]);

explosions.splice(i, 1);

}

for (i = 0; i < playerBullets.length; i++)

{

removeChild(playerBullets[i]);

playerBullets.splice(i, 1);

}

for (i = 0; i < powers.length; i++)

{

removeChild(powers[i]);

powers.splice(i, 1);

}

}

//remove all event listeners

enemyTimer.removeEventListener(TimerEvent.TIMER, spawnEnemy);

stage.removeEventListener(KeyboardEvent.KEY\_DOWN, aKeyPressed);

stage.removeEventListener(KeyboardEvent.KEY\_UP, aKeyReleased);

removeEventListener(Event.ENTER\_FRAME, animate);

shootingTimer.removeEventListener(TimerEvent.TIMER, checkShoot);

explosionTimer.removeEventListener(TimerEvent.TIMER, removeExplosion);

playerHitTimer.removeEventListener(TimerEvent.TIMER, removePlayerHit);

spawnPowerUpTimer.removeEventListener(TimerEvent.TIMER, spawnPowers);

powerUpTimer.removeEventListener(TimerEvent.TIMER, removePowers);

effectsChannel = gameOverSound.play();

//change scene to end frame

gotoAndPlay("end");

}

function removeExplosion(e: TimerEvent): void

{

//remove explosions when timer ticks

for (var i: uint = 0; i < explosions.length; i++)

{

removeChild(explosions[i]);

explosions.splice(i, 1);

}

}

function spawnExplosion(enemy: MovieClip): void

{

//creating and spawning explosion on to stage and adding to explosions array

var explosion: MovieClip = new Explosion();

addChildAt(explosion, numChildren - 1);

explosions.push(explosion);

//setting explosion position to enemy position

explosion.x = enemy.x;

explosion.y = enemy.y;

explosion.rotation = enemy.rotation;

explosionTimer.reset();

explosionTimer.start();

effectsChannel = explosionSound.play();

}

function playerEnemyHitTest(): void

{

for (var i: uint = 0; i < enemies.length; i++)

{

//checking if player hits an enemy

if (player.hitZone.hitTestObject(enemies[i]))

{

//check if shield is active

if (player.shield == true)

{

score += (enemies[i].pointVal) / 2;

spawnExplosion(enemies[i]);

removeChild(enemies[i]);

enemies.splice(i, 1);

effectsChannel = playerHitShieldSound.play();

}

else

{

lives -= 1;

removeChild(enemies[i]);

enemies.splice(i, 1);

playerHitAnimation();

effectsChannel = playerHitSound.play();

}

}

}

}

function playerHitAnimation(): void

{

//show player hit image

playerHit.visible = true;

playerHitTimer.start()

}

function removePlayerHit(e: TimerEvent): void

{

//remove player hit image

if (playerHit.visible)

{

playerHit.visible = false;

playerHitTimer.stop();

playerHitTimer.reset();

}

}

function bulletEnemyHitTest(): void

{

for (var i: uint = 0; i < enemies.length; i++)

{

for (var j: uint = 0; j < playerBullets.length; j++)

{

//checking if a bullet hits an enemy

if (playerBullets[j].hitTestObject(enemies[i]))

{

//remove 1 health

enemies[i].healthVal -= 1;

//check if health is 0 or less

if (enemies[i].healthVal <= 0)

{

//increase score

score += enemies[i].pointVal;

spawnExplosion(enemies[i]);

removeChild(enemies[i]);

enemies.splice(i, 1);

}

removeChild(playerBullets[j]);

playerBullets.splice(j, 1);

break;

}

}

}

}

function moveEnemy(): void

{

//move each enemy in enemies array

for (var i: uint = 0; i < enemies.length; i++)

{

//get angle between enemy's current position and player's current position

var deg = getAngle(enemies[i].x, enemies[i].y, player.x, player.y);

enemies[i].rotation = deg;

var rad: Number = degToRad(enemies[i].rotation)

//move towards player's current position

enemies[i].x += Math.cos(rad) \* enemies[i].speedVal;

enemies[i].y += Math.sin(rad) \* enemies[i].speedVal;

}

}

function spawnEnemy(e: TimerEvent): void

{

//creating and spawning enemy on to stage and adding to enemies array

var enemy: MovieClip = new Enemies();

addChildAt(enemy, numChildren - 1);

enemies.push(enemy);

//Picking Spawn Location

var spawnLocation: Number = Math.random()

if (spawnLocation <= 0.25)

{

enemy.x = stage.stageWidth \* Math.random();

enemy.y = 0;

}

else if (spawnLocation <= 0.50)

{

enemy.x = stage.stageWidth \* Math.random();

enemy.y = stage.stageHeight;

}

else if (spawnLocation <= 0.75)

{

enemy.x = 0;

enemy.y = stage.stageHeight \* Math.random();

}

else

{

enemy.x = stage.stageWidth;

enemy.y = stage.stageHeight \* Math.random();

}

//Picking Enemy

var pickEnemy: int;

if (score < LEVEL\_1)

{

pickEnemy = Math.ceil(Math.random() \* 3);

}

else if (score < LEVEL\_2)

{

pickEnemy = Math.ceil(Math.random() \* 6);

}

else if (score < LEVEL\_3)

{

pickEnemy = Math.ceil(Math.random() \* 9);

}

else if (score > LEVEL\_BOSS)

{

pickEnemy = Math.ceil(Math.random() \* 10);

}

else

{

pickEnemy = Math.ceil(Math.random() \* 9);

}

if (pickEnemy == 1)

{

enemy.gotoAndStop("1 1");

enemy.pointVal = 1;

enemy.healthVal = 1;

enemy.speedVal = ENEMY\_BASESPEED \* 1;

}

else if (pickEnemy == 2)

{

enemy.gotoAndStop("1 2");

enemy.pointVal = 2;

enemy.healthVal = 1;

enemy.speedVal = ENEMY\_BASESPEED \* 1;

}

else if (pickEnemy == 3)

{

enemy.gotoAndStop("1 3");

enemy.pointVal = 5;

enemy.healthVal = 1;

enemy.speedVal = ENEMY\_BASESPEED \* 1;

}

else if (pickEnemy == 4)

{

enemy.gotoAndStop("2 1");

enemy.pointVal = 5;

enemy.healthVal = 2;

enemy.speedVal = ENEMY\_BASESPEED \* 1.25;

}

else if (pickEnemy == 5)

{

enemy.gotoAndStop("2 2");

enemy.pointVal = 10;

enemy.healthVal = 1;

enemy.speedVal = ENEMY\_BASESPEED \* 1.5;

}

else if (pickEnemy == 6)

{

enemy.gotoAndStop("2 3");

enemy.pointVal = 15;

enemy.healthVal = 2;

enemy.speedVal = ENEMY\_BASESPEED \* 1.25;

}

else if (pickEnemy == 7)

{

enemy.gotoAndStop("3 1");

enemy.pointVal = 10 + Math.ceil(Math.random() \* 10);

enemy.healthVal = 3;

enemy.speedVal = ENEMY\_BASESPEED \* 1;

}

else if (pickEnemy == 8)

{

enemy.gotoAndStop("3 2");

enemy.pointVal = 10 + Math.ceil(Math.random() \* 15);

enemy.healthVal = 3;

enemy.speedVal = ENEMY\_BASESPEED \* 1;

}

else if (pickEnemy == 9)

{

enemy.gotoAndStop("3 3");

enemy.pointVal = 10 + Math.ceil(Math.random() \* 20);

enemy.healthVal = 3;

enemy.speedVal = ENEMY\_BASESPEED \* 1;

}

else if (pickEnemy == 10)

{

enemy.gotoAndStop("4 1");

enemy.pointVal = 50;

enemy.healthVal = 10;

enemy.speedVal = ENEMY\_BASESPEED \* .5;

}

}

function checkShoot(e: TimerEvent): void

{

//when timer ticks, enables player to shoot again

isShooting = false;

}

function shootPlayer(): void

{

//check if it's shooting

if (!isShooting)

{

//Shoot

spawnBullet();

isShooting = true;

}

}

function moveBullet(): void

{

//move all bullets in bullets array

for (var i: uint = 0; i < playerBullets.length; i++)

{

var rad: Number = degToRad(playerBullets[i].rotation);

playerBullets[i].x += Math.sin(rad) \* PLAYER\_BULLETSPEED;

playerBullets[i].y -= Math.cos(rad) \* PLAYER\_BULLETSPEED;

//check if bullet is off stage

if (playerBullets[i].y > stage.stageHeight + BOUNDARY)

{

removeChild(playerBullets[i]);

playerBullets.splice(i, 1);

}

else if (playerBullets[i].y < 0 - BOUNDARY)

{

removeChild(playerBullets[i]);

playerBullets.splice(i, 1);

}

else if (playerBullets[i].x < 0 - BOUNDARY)

{

removeChild(playerBullets[i]);

playerBullets.splice(i, 1);

}

else if (playerBullets[i].x > stage.stageWidth + BOUNDARY)

{

removeChild(playerBullets[i]);

playerBullets.splice(i, 1);

}

}

}

function spawnBullet(): void

{

//creating and spawning bullet on to stage and adding to bullets array

var bullet: MovieClip = new PlayerBullet();

addChildAt(bullet, numChildren - 1);

playerBullets.push(bullet);

//setting position and rotation to player's

bullet.x = player.x;

bullet.y = player.y;

bullet.rotation = player.rotation;

effectsChannel = playerShootSound.play();

}

function rotatePlayer(): void

{

//rotate player according to key press and shoot bullet in direction player is facing

if (upPressed)

{

player.rotation = 0;

shootPlayer();

}

if (leftPressed)

{

player.rotation = 270;

shootPlayer();

}

if (downPressed)

{

player.rotation = 180;

shootPlayer();

}

if (rightPressed)

{

player.rotation = 90;

shootPlayer();

}

}

function movePlayer(): void

{

//move player in corrisponding direction to key press, rotate in direction

if (aPressed)

{

player.x -= playerXSpeed;

player.rotation = 270;

}

if (dPressed)

{

player.x += playerXSpeed;

player.rotation = 90;

}

if (wPressed)

{

player.y -= playerYSpeed;

player.rotation = 0;

}

if (sPressed)

{

player.y += playerYSpeed;

player.rotation = 180;

}

//check if player is staying within stage borders

if (player.x < 0)

{

player.x = 0;

}

else if (player.x > stage.stageWidth)

{

player.x = stage.stageWidth;

}

if (player.y < 0)

{

player.y = 0;

}

else if (player.y > stage.stageHeight)

{

player.y = stage.stageHeight;

}

}

function aKeyPressed(e: KeyboardEvent): void

{

//check when key is being pressed

if (e.keyCode == Keyboard.LEFT)

{

leftPressed = true;

}

if (e.keyCode == Keyboard.RIGHT)

{

rightPressed = true;

}

if (e.keyCode == Keyboard.UP)

{

upPressed = true;

}

if (e.keyCode == Keyboard.DOWN)

{

downPressed = true;

}

if (e.keyCode == Keyboard.W)

{

wPressed = true;

}

if (e.keyCode == Keyboard.A)

{

aPressed = true;

}

if (e.keyCode == Keyboard.S)

{

sPressed = true;

}

if (e.keyCode == Keyboard.D)

{

dPressed = true;

}

}

function aKeyReleased(e: KeyboardEvent): void

{

//check when key is being released

if (e.keyCode == Keyboard.LEFT)

{

leftPressed = false;

}

if (e.keyCode == Keyboard.RIGHT)

{

rightPressed = false;

}

if (e.keyCode == Keyboard.UP)

{

upPressed = false;

}

if (e.keyCode == Keyboard.DOWN)

{

downPressed = false;

}

if (e.keyCode == Keyboard.W)

{

wPressed = false;

}

if (e.keyCode == Keyboard.A)

{

aPressed = false;

}

if (e.keyCode == Keyboard.S)

{

sPressed = false;

}

if (e.keyCode == Keyboard.D)

{

dPressed = false;

}

}

function degToRad(deg: Number): Number

{

//convert degrees to radians

return deg \* Math.PI / 180;

}

function radToDeg(rad: Number): Number

{

//convert radians to degrees

return rad \* 180 / Math.PI;

}

function getAngle(x1: Number, y1: Number, x2: Number, y2: Number): Number

{

//getting angle between object1 and object2

var rad: Number = Math.atan2(y2 - y1, x2 - x1);

var deg: Number = radToDeg(rad);

return deg;

}

function update(): void

{

//setting lives and score text boxes to corrisponding values

livesTxt.text = String(lives);

scoreTxt.text = String(score);

//setting game over is lives less than or equal to 0

if (lives <= 0)

{

gameOver();

}

}