using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Domagala

public class AdjustVolume : MonoBehaviour

{

public void ChangeVol(float newValue)

{

// DEFINING NEW VOLUME (EQUAL TO INITIAL VOLUME OF AUDIOLISTENER

float newVol = AudioListener.volume;

// VOLUME VALUE CHANGED AS SLIDE IS MOVED

newVol = newValue;

// NEW VOLUME SENT TO AUDIO LISTENER

AudioListener.volume = newVol;

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class ContinuousAudio : MonoBehaviour

{

AudioSource audioSource;

void Awake()

{

DontDestroyOnLoad(transform.gameObject);

audioSource = GetComponent<AudioSource>();

}

public void play()

{

if (audioSource.isPlaying)

{

return;

}

audioSource.Play();

}

public void stop()

{

audioSource.Stop();

}

}

using UnityEngine;

using UnityEngine.UI;

// Author: Alex Domagala

public class SoundActivation : MonoBehaviour

{

public Button Play, Options, Quit, Back; // References to all buttons that produce a "click noise"

void Start()

{

Play.onClick.AddListener(TaskOnClick);

Options.onClick.AddListener(TaskOnClick);

Quit.onClick.AddListener(TaskOnClick);

Back.onClick.AddListener(TaskOnClick);

}

void TaskOnClick()

{

AudioSource audio = GetComponent<AudioSource>();

audio.Play();

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: David Nygren

public class ConstantScroll : MonoBehaviour

{

public float bgSpeed;

public Renderer bgRend;

void Update()

{

// MOVEMENT OF BACKGROUND GRAPHIC

bgRend.material.mainTextureOffset += new Vector2(bgSpeed \* Time.deltaTime, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class Main : MonoBehaviour

{

Rigidbody2D player;

SpriteRenderer sprite;

Animator animator;

Vector2 directionedMovement;

Vector2 userInput;

GameObject quad;

ConstantScroll quadScript;

GameObject ground;

BoxCollider2D groundCollider;

GameObject MovingParticlesLeft;

SpriteRenderer particleSpriteLeft;

Animator particleAnimationLeft;

GameObject MovingParticlesRight;

SpriteRenderer particleSpriteRight;

Animator particleAnimationRight;

bool hasStarted;

bool hasEnded;

bool slowDown;

bool heldDirection;

bool inAir;

bool jump;

bool right;

bool left;

int jumpCounter;

float speedMultiplier;

void Start()

{

GameObject.FindGameObjectWithTag("Audio").GetComponent<ContinuousAudio>().play();

player = GetComponent<Rigidbody2D>();

sprite = GetComponent<SpriteRenderer>();

animator = GetComponent<Animator>();

userInput = new Vector2(0f, 0f);

quad = GameObject.Find("Quad");

quadScript = quad.GetComponent<ConstantScroll>();

ground = GameObject.Find("Barriers/Ground");

groundCollider = ground.GetComponent<BoxCollider2D>();

MovingParticlesLeft = GameObject.Find("Ball/MovingParticlesLeft");

particleSpriteLeft = MovingParticlesLeft.GetComponent<SpriteRenderer>();

particleSpriteLeft.enabled = false;

particleAnimationLeft = MovingParticlesLeft.GetComponent<Animator>();

MovingParticlesRight = GameObject.Find("Ball/MovingParticlesRight");

particleSpriteRight = MovingParticlesRight.GetComponent<SpriteRenderer>();

particleSpriteRight.enabled = false;

particleAnimationRight = MovingParticlesRight.GetComponent<Animator>();

heldDirection = true;

jumpCounter = 0;

speedMultiplier = 1f;

}

void Update()

{

// CHECKS IF THE GAME HAS ENDED

if (hasEnded)

{

return;

}

// CHECKS IF THE PLAYER IS GROUNDED

if (player.IsTouching(groundCollider))

{

inAir = false;

}

else

{

inAir = true;

}

//BASIC HORIZONTAL MOVEMENT INPUT

Vector2 userInput = new Vector2(Input.GetAxisRaw("Horizontal"), 0f);

directionedMovement = userInput.normalized \* 0.5f;

// CHECKS IF THE GAME HAS STARTED

if (!hasStarted)

{

// STARTS THE QUAD MOVEMENT WHEN THE GAME BEGINS

if (directionedMovement.x != 0)

{

hasStarted = true;

GameObject.Find("Tutorial").SetActive(false);

quadScript.bgSpeed = 0.1f \* speedMultiplier;

}

else

{

return;

}

}

// PLAYER ANIMATION CONTROL

if ((userInput.x > 0) && (heldDirection = true))

{

sprite.flipX = true;

particleSpriteLeft.enabled = true;

particleSpriteRight.enabled = false;

heldDirection = false;

}

else if ((userInput.x < 0) && (!heldDirection))

{

sprite.flipX = false;

particleSpriteLeft.enabled = false;

particleSpriteRight.enabled = true;

heldDirection = true;

}

animator.SetFloat("xSpeed", Mathf.Abs(directionedMovement.x));

// ADDITIONAL Y-AXIS GRAVITY FORCE

float multiplier = 0f;

if (inAir)

{

multiplier = -2f;

// CHECKS IF THE PLAYER HAS ACTIVATED SLOW DOWN

if (Input.GetKey(KeyCode.LeftShift) || Input.GetKey(KeyCode.DownArrow) || Input.GetKey("s"))

{

// ADJUSTS QUAD MOVEMENT SPEED

quadScript.bgSpeed = 0.05f \* speedMultiplier;

slowDown = true;

// REDUCES PLAYER FALLING SPEED

multiplier = -.75f;

}

else

{

// REGULAR QUAD MOVEMENT SPEED

quadScript.bgSpeed = 0.1f \* speedMultiplier;

slowDown = false;

}

}

else

{

// REGULAR QUAD MOVEMENT SPEED

quadScript.bgSpeed = 0.1f \* speedMultiplier;

slowDown = false;

}

// ASSIGNS Y-AXIS FORCE

directionedMovement.y = multiplier;

// PLAYER MOVEMENT PARTICLE CONTROL

if (directionedMovement.x == 0)

{

animator.enabled = false;

if (sprite.flipX)

{

particleSpriteLeft.enabled = false;

}

else

{

particleSpriteRight.enabled = false;

}

if (!inAir) {

// X-AXIS RESISTANCE FORCE

directionedMovement.x = -1.25f \* speedMultiplier;

}

}

else

{

animator.enabled = true;

if (inAir)

{

particleSpriteLeft.enabled = false;

particleSpriteRight.enabled = false;

}

else if (sprite.flipX)

{

particleSpriteLeft.enabled = true;

}

else

{

particleSpriteRight.enabled = true;

// X-AXIS RESISTANCE FORCE

directionedMovement.x = -1.75f \* speedMultiplier;

}

}

// JUMP CONTROL

if ((Input.GetKeyDown(KeyCode.UpArrow)|| Input.GetKeyDown("w") || Input.GetKeyDown("space")) && !inAir)

{

jump = true;

// CHECKS WHAT DIRECTION PLAYER IS FACING IN AIR

if (directionedMovement.x > 0)

{

right = true;

}

else if (directionedMovement.x < 0)

{

left = true;

}

}

Debug.Log("X: " + directionedMovement.x);

Debug.Log("Y: " + directionedMovement.y);

}

void FixedUpdate()

{

// DETERMINES IF PLAYER IS JUMPING OR MOVING

if (jump)

{

jumpMove();

}

else if (!hasEnded)

{

player.MovePosition(player.position + directionedMovement);

}

}

void jumpMove()

{

// JUMP CONTROLS MOVEMENT SCRIPT FOR 15 FRAMES

if (jumpCounter == 15)

{

// RESETS JUMP PREFERENCES

jump = false;

right = false;

left = false;

jumpCounter = 0;

}

else

{

jumpCounter++;

// ANGLED JUMPING AND NORMAL JUMPING

if (right)

{

// ANGLED JUMPING TO THE RIGHT

player.MovePosition(player.position + (new Vector2(1f, (15 - jumpCounter))));

}

else if (left)

{

// ANGLED JUMPING TO THE LEFT

player.MovePosition(player.position + (new Vector2(-1f \* speedMultiplier, (15 - jumpCounter))));

}

else

{

// NORMAL JUMPING

player.MovePosition(player.position + (new Vector2(0f, (15 - jumpCounter))));

}

}

}

public bool getSlowDown()

{

return slowDown;

}

public bool getHasStarted()

{

return hasStarted;

}

public bool getHasEnded()

{

return hasEnded;

}

public void setHasEnded()

{

hasEnded = true;

quadScript.bgSpeed = 0f;

}

public void setSpeedMultiplier(float speedMultiplier)

{

this.speedMultiplier = speedMultiplier;

}

public void removeBall()

{

// DISABLES BALL SPRITERENDERER AND MOVEMENT PARTICLES

sprite.enabled = false;

particleSpriteLeft.enabled = false;

particleSpriteRight.enabled = false;

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class Obstacle : MonoBehaviour

{

GameObject ball;

Main ballMovement;

CircleCollider2D ballCollider;

Rigidbody2D obstacle;

BoxCollider2D obstacleCollider;

float speedMultiplier;

bool hasStarted;

bool hasEnded;

bool gameOver;

bool slowDown;

void Start()

{

ball = GameObject.Find("Ball");

ballMovement = ball.GetComponent<Main>();

ballCollider = ball.GetComponent<CircleCollider2D>();

obstacle = GetComponent<Rigidbody2D>();

obstacleCollider = GetComponent<BoxCollider2D>();

speedMultiplier = 1f;

hasStarted = false;

hasEnded = false;

gameOver = false;

}

void Update()

{

// CHECKS IF THE GAME HAS STARTED

if (!hasStarted)

{

// UPDATES hasStarted

hasStarted = ballMovement.getHasStarted();

return;

}

// UPDATES hasEnded

hasEnded = ballMovement.getHasEnded();

// CHECKS IF THE GAME IS OVER

if (hasEnded)

{

if (!gameOver)

{

GameObject.Find("GameOver").GetComponent<GameOver>().end();

gameOver = true;

}

return;

}

// UPDATES speedMultiplier

speedMultiplier = ball.GetComponentInChildren<Score>().getSpeedMultiplier();

// CHECKS IF THE PLAYER HAS ACTIVATED SLOW DOWN

if (ballMovement.getSlowDown())

{

// MOVES THE OBSTACLES AT A SLOWER RATE

obstacle.MovePosition(obstacle.position + new Vector2(-0.62f \* speedMultiplier, 0f));

}

else

{

// MOVES THE OBSTACLES AT THE REGULAR RATE

obstacle.MovePosition(obstacle.position + new Vector2(-1.27f \* speedMultiplier, 0f));

}

// CHECKS IF THE PLAYER HAS COLLIDED WITH AN OBSTACLE

if (obstacleCollider.IsTouching(ballCollider))

{

// UPDATES hasEnded

hasEnded = true;

// UPDATES hasEnded IN THE MOVEMENT CLASS OF THE BALL OBJECT

ballMovement.setHasEnded();

// REMOVES BALL FROM THE SCREEN

ballMovement.removeBall();

}

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class Obstacle\_Controller : MonoBehaviour

{

void OnTriggerEnter2D(Collider2D col)

{

// CHECKS IF THE COLLISION IS WITH THE LEFT BOUND

if (col.gameObject.tag == "Obstacles")

{

// SELECTS RANDOM Y COORDINATE TO RESET OBSTACLE TO

float random = Random.Range(-32, 40);

// RESETS POSITION OF OBSTACLE

col.gameObject.transform.position = new Vector3(0f, random, 2f);

}

// MAKES THE OBSTACLE SCOREABLE AGAIN

ScoreUpdate scoreUpdate = col.gameObject.GetComponentInChildren<ScoreUpdate>();

scoreUpdate.resetScored();

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class Score : MonoBehaviour

{

GameObject ball;

Main ballMovement;

int scoreThreshold;

float speedMultiplier;

bool thresholdReached;

void Start()

{

ball = GameObject.Find("Ball");

ballMovement = ball.GetComponent<Main>();

scoreThreshold = 10; // Increments at which game speed increases

speedMultiplier = 1f;

thresholdReached = false; // Tracks whether the player has passsed the current obstacle

}

void Update()

{

int score = ScoreDisplay.scoreValue;

// CHECKS IF THE PLAYER'S SCORE IS A MULTIPLE OF 10 AND THAT THE CODE IS RAN ONLY ONCE

if (score != 0 && (score % scoreThreshold == 0) && !thresholdReached)

{

// INCREASES QUAD/OBSTACLE MOVEMENT

thresholdReached = true;

speedMultiplier += 0.2f;

ballMovement.setSpeedMultiplier(speedMultiplier);

}

else if (score % scoreThreshold != 0)

{

thresholdReached = false;

}

Debug.Log("Speed Multiplier: " + speedMultiplier);

}

public void increaseScore(int increment)

{

// INCREMENTS GLOBAL SCORE VARIABLE

ScoreDisplay.scoreValue += increment;

}

public float getSpeedMultiplier()

{

return speedMultiplier;

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class ScoreUpdate : MonoBehaviour

{

GameObject ball;

CircleCollider2D ballCollider;

Score score;

BoxCollider2D scoreThreshold;

bool scored;

void Start()

{

ball = GameObject.Find("Ball");

ballCollider = ball.GetComponent<CircleCollider2D>();

score = ball.GetComponent<Score>();

scoreThreshold = GetComponent<BoxCollider2D>();

scored = false; // Tracks whether the player has passsed the current obstacle

}

void Update()

{

// CHECKS IF THE PLAYER PASSES AN OBSTACLE

if (scoreThreshold.IsTouching(ballCollider) && !scored)

{

score.increaseScore(1);

scored = true;

}

}

public void resetScored()

{

scored = false;

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class GameOver : MonoBehaviour

{

HighScoreDisplay highScore;

GameObject gameOver;

void Start()

{

highScore = GetComponentInChildren<HighScoreDisplay>();

gameOver = GameObject.Find("GameOver/Canvas");

gameOver.SetActive(false);

}

public void end ()

{

// ENABLES GAME OVER SCREEN CANVAS

gameOver.SetActive(true);

bool newRecord = false;

// CHECKS IF A NEW AND NON-ZERO HIGH SCORE HAS BEEN REACHED

if (ScoreDisplay.scoreValue != 0 && HighScoreDisplay.highScoreValue < ScoreDisplay.scoreValue)

{

// UPDATES HIGH SCORE

HighScoreDisplay.highScoreValue = ScoreDisplay.scoreValue;

newRecord = true;

}

// DISPLAYS HIGH SCORE

highScore.display(newRecord);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

// Author: Alex Huang

public class HighScoreDisplay : MonoBehaviour

{

public static int highScoreValue = 0;

Text score;

newDisplay newIndicator;

void Start()

{

score = GetComponent<Text>();

newIndicator = GetComponentInChildren<newDisplay>();

}

public void display(bool newRecord)

{

// CHECKS IF A NEW RECORD HAS BEEN REACHED

if (newRecord)

{

newIndicator.appear();

}

score.text = "High Score: " + highScoreValue;

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

// Author: Alex Huang

public class newDisplay : MonoBehaviour

{

Text text;

void Start()

{

text = GetComponent<Text>();

text.enabled = false;

}

public void appear()

{

text.enabled = true;

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

// Author: David Nygren

public class newScene : MonoBehaviour

{

public void changeScene(string scenename)

{

SceneManager.LoadScene(scenename);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

// Author: David Nygren

public class PauseMenu : MonoBehaviour

{

public static bool GameIsPaused = false;

public GameObject pauseMenuUI;

void Update()

{

if (Input.GetKeyDown(KeyCode.Escape))

{

if (GameIsPaused)

{

Resume();

}

else

{

Pause();

}

}

}

public void Resume()

{

pauseMenuUI.SetActive(false);

Time.timeScale = 1f;

GameIsPaused = false;

}

void Pause()

{

pauseMenuUI.SetActive(true);

Time.timeScale = 0f;

GameIsPaused = true;

}

public void LoadMenu()

{

Time.timeScale = 1f;

GameObject.FindGameObjectWithTag("Audio").GetComponent<ContinuousAudio>().stop();

SceneManager.LoadScene("MainMenu");

}

public void Restart()

{

Time.timeScale = 1f;

SceneManager.LoadScene("Game");

}

public void QuitGame()

{

Application.Quit();

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

// Author: Alex Huang

public class ScoreDisplay : MonoBehaviour

{

public static int scoreValue;

Text score;

void Start()

{

scoreValue = 0;

score = GetComponent<Text>();

}

void Update()

{

score.text = "Score: " + scoreValue;

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class OptionsNormalization : MonoBehaviour

{

RectTransform button;

void Start()

{

button = GetComponent<RectTransform>();

float newDistanceX = 0.69f \* ((float)Screen.width / 2f);

float newDistanceY = 0.027f \* ((float)Screen.height / 2f);

button.anchoredPosition = new Vector3(-newDistanceX, -newDistanceY, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class PlayNormalization : MonoBehaviour

{

RectTransform button;

void Start()

{

button = GetComponent<RectTransform>();

float newDistanceX = 0.019f \* ((float)Screen.width / 2f);

float newDistanceY = 0.5f \* ((float)Screen.height / 2f);

button.anchoredPosition = new Vector3(-newDistanceX, -newDistanceY, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class QuitNormalization : MonoBehaviour

{

RectTransform button;

void Start()

{

button = GetComponent<RectTransform>();

float newDistanceX = 0.6f \* ((float)Screen.width / 2f);

float newDistanceY = 0.027f \* ((float)Screen.height / 2f);

button.anchoredPosition = new Vector3(newDistanceX, -newDistanceY, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class SpherusBallNormalization : MonoBehaviour

{

RectTransform image;

int keyWidth = 1149;

int keyHeight = 591;

void Start()

{

image = GetComponent<RectTransform>();

float widthMultiplier = (float)Screen.width / (float)keyWidth;

float heightMultiplier = (float)Screen.height / (float)keyHeight;

image.transform.localScale = new Vector3(widthMultiplier, heightMultiplier, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class XSNormalization : MonoBehaviour

{

RectTransform image;

int keyWidth = 1149;

int keyHeight = 591;

void Start()

{

image = GetComponent<RectTransform>();

float widthMultiplier = (float)Screen.width / (float)keyWidth;

float heightMultiplier = (float)Screen.height / (float)keyHeight;

image.transform.localScale = new Vector3(0.75f \* widthMultiplier, 0.75f \* heightMultiplier, 0f);

float newDistanceX = 0.005f \* ((float)Screen.width / 2f);

float newDistanceY = 0.72f \* ((float)Screen.height / 2f);

image.anchoredPosition = new Vector3(-newDistanceX, -newDistanceY, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class BackNormalization : MonoBehaviour

{

RectTransform button;

void Start()

{

button = GetComponent<RectTransform>();

float newDistanceY = 0.41f \* ((float)Screen.height / 2f);

button.anchoredPosition = new Vector3(0f, -newDistanceY, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class FullscreenNormalization : MonoBehaviour

{

RectTransform button;

void Start()

{

button = GetComponent<RectTransform>();

float newDistanceX = 0.65f \* ((float)Screen.width / 2f);

float newDistanceY = 0.23f \* ((float)Screen.height / 2f);

button.anchoredPosition = new Vector3(newDistanceX, newDistanceY, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class GraphicsNormalization : MonoBehaviour

{

RectTransform button;

void Start()

{

button = GetComponent<RectTransform>();

float newDistanceX = 0.68f \* ((float)Screen.width / 2f);

float newDistanceY = 0.23f \* ((float)Screen.height / 2f);

button.anchoredPosition = new Vector3(-newDistanceX, -newDistanceY, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class ResolutionNormalization : MonoBehaviour

{

RectTransform button;

void Start()

{

button = GetComponent<RectTransform>();

float newDistanceX = 0.65f \* ((float)Screen.width / 2f);

float newDistanceY = 0.26f \* ((float)Screen.height / 2f);

button.anchoredPosition = new Vector3(-newDistanceX, newDistanceY, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class SliderNormalization : MonoBehaviour

{

RectTransform button;

void Start()

{

button = GetComponent<RectTransform>();

float newDistanceX = 0.66f \* ((float)Screen.width / 2f);

float newDistanceY = 0.26f \* ((float)Screen.height / 2f);

button.anchoredPosition = new Vector3(newDistanceX, -newDistanceY, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class TextNormalization : MonoBehaviour

{

RectTransform button;

void Start()

{

button = GetComponent<RectTransform>();

float newDistanceY = 0.52f \* ((float)Screen.height / 2f);

button.anchoredPosition = new Vector3(0f, newDistanceY, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class VolumeNormalization : MonoBehaviour

{

RectTransform button;

void Start()

{

button = GetComponent<RectTransform>();

float newDistanceX = 0.55f \* ((float)Screen.width / 2f);

float newDistanceY = 0.15f \* ((float)Screen.height / 2f);

button.anchoredPosition = new Vector3(newDistanceX, -newDistanceY, 0f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class CameraNormalization : MonoBehaviour

{

void Start()

{

float targetaspect = 16.0f / 9.0f;

float windowaspect = (float)Screen.width / (float)Screen.height;

// CONVERSION RATIO

float scaleheight = windowaspect / targetaspect;

Camera camera = GetComponent<Camera>();

if (scaleheight < 1.0f)

{

Rect rect = camera.rect;

rect.width = 1.0f;

rect.height = scaleheight;

rect.x = 0;

rect.y = (1.0f - scaleheight) / 2.0f;

camera.rect = rect;

}

else

{

float scalewidth = 1.0f / scaleheight;

Rect rect = camera.rect;

rect.width = scalewidth;

rect.height = 1.0f;

rect.x = (1.0f - scalewidth) / 2.0f;

rect.y = 0;

camera.rect = rect;

}

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class GeneralNormalization : MonoBehaviour

{

RectTransform rectTrans;

int keyWidth = 1272;

int keyHeight = 591;

void Start()

{

rectTrans = GetComponent<RectTransform>();

float widthMultiplier = (float)Screen.width / (float)keyWidth;

float heightMultiplier = (float)Screen.height / (float)keyHeight;

rectTrans.transform.localScale = new Vector3(widthMultiplier, heightMultiplier, 1f);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Author: Alex Huang

public class ScoreNormalization : MonoBehaviour

{

RectTransform text;

void Start()

{

text = GetComponent<RectTransform>();

float newDistanceX = 0.38f \* ((float)Screen.width / 2f);

float newDistanceY = 0.63f \* ((float)Screen.height / 2f);

text.anchoredPosition = new Vector3(-newDistanceX, newDistanceY, 0f);

}

}