**Individual Project 3**

**Spider Smash**

By: Nicholas Ryder

Prof. Alrajab

5/4/2020

**Overview:**

Theme/ Setting/ Genre:

* The Theme is a Carnival Game

Gameplay Mechanics:

* VR Gameplay
* Collision between spider and hammer adds points
* Collision between floor and toilet paper starts game
* Game Over when Timer hits 0
* Highscore Saving

**Explanation:**

I made a simple VR game using assets provided by Valve. It is a Carnival/Arcade game based on whack a mole. To start the game you need to take the cube on the platform to the left of the table and throw it at the stack of toilet paper once the game starts you use the mallet placed at the front of the game table to hit the spiders that pop up each spider hit adds a point to the score. When the timer hits zero the game will end if your score is higher than the high score, your score will over right the high score and will be carried over when you start up the game again until it is beaten.

**Scripts:**

**GameController:**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class GameController : MonoBehaviour

{

public TextMesh timerText;

public float gameTimer = 60f;

public TextMesh gameName;

public TextMesh Score;

public static bool Started;

public TextMesh Instruct;

public TextMesh High;

public GameObject Hammer;

public GameObject spiderContainer;

private Spider[] spiders;

public float showSpiderTime = 1.5f;

// Start is called before the first frame update

void Start()

{

spiders = spiderContainer.GetComponentsInChildren<Spider>();

High.text = "High Score: " + PlayerPrefs.GetFloat("HighScore", 0).ToString();

}

// Update is called once per frame

void Update()

{

if (Started == true)

{

gameTimer -= Time.deltaTime;

Instruct.text = "GAME STARTED";

if (gameTimer > 0f)

{

Score.text = "" + OnCollisionSpider.points;

timerText.text = "" + Mathf.Floor(gameTimer);

showSpiderTime -= Time.deltaTime;

if (showSpiderTime < 0f)

{

spiders[Random.Range(0, spiders.Length)].ShowSpider();

showSpiderTime = 1.5f;

}

;

}

else

{

gameName.text = "GAME OVER";

Instruct.text = "To Reset Game Place the Hammer\n on the Platform to the left of the Game Table.";

if(OnCollisionSpider.points > PlayerPrefs.GetFloat("HighScore", 0))

{

PlayerPrefs.SetFloat("HighScore", OnCollisionSpider.points);

High.text = "High Score: " + OnCollisionSpider.points;

}

}

}

}

}

**Spider:**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Spider : MonoBehaviour

{

public float visibleYHeight = 17.22732f;

public float hiddenYHeight = 17.109f;

private Vector3 myNewXYZPosition;

public float speed = 4f;

public float hideSpiderTimer = 1.5f;

void Awake()

{

HideSpider();

transform.localPosition = myNewXYZPosition;

}

void Update()

{

transform.localPosition = Vector3.Lerp(transform.localPosition, myNewXYZPosition, Time.deltaTime \* speed);

hideSpiderTimer -= Time.deltaTime;

if(hideSpiderTimer < 0)

{

HideSpider();

}

}

public void HideSpider()

{

myNewXYZPosition = new Vector3(transform.localPosition.x, hiddenYHeight, transform.localPosition.z);

}

public void ShowSpider()

{

myNewXYZPosition = new Vector3(transform.localPosition.x, visibleYHeight,transform.localPosition.z);

hideSpiderTimer = 1.5f;

}

}

**OnCollisionSpider:**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class OnCollisionSpider : MonoBehaviour

{

public Spider \_SpiderHider;

public Spider \_SpiderHider2;

public Spider \_SpiderHider3;

public Spider \_SpiderHider4;

public Spider \_SpiderHider5;

public Spider \_SpiderHider6;

public Spider \_SpiderHider7;

public Spider \_SpiderHider8;

public Spider \_SpiderHider9;

public Rigidbody Hammer;

public static float points = 0;

void Start()

{

Hammer = GetComponent<Rigidbody>();

}

void Update()

{

Hammer.detectCollisions = true;

}

void OnCollisionEnter(Collision spider)

{

if (spider.gameObject.tag == "Spider")

{

points++;

\_SpiderHider.HideSpider();

\_SpiderHider2.HideSpider();

\_SpiderHider3.HideSpider();

\_SpiderHider4.HideSpider();

\_SpiderHider5.HideSpider();

\_SpiderHider6.HideSpider();

\_SpiderHider7.HideSpider();

\_SpiderHider8.HideSpider();

\_SpiderHider9.HideSpider();

}

}

}

**OnCollisionTable:**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class OnCollisionTable : MonoBehaviour

{

public GameObject Hammer;

// Start is called before the first frame update

void Start()

{

}

// Update is called once per frame

void Update()

{

}

void OnCollisionEnter(Collision Table)

{

if(Table.gameObject.tag == "Hammer")

{

Destroy(Hammer);

SceneManager.LoadScene("Spider Smash");

Valve.VR.OpenVR.System.ResetSeatedZeroPose();

Valve.VR.OpenVR.Compositor.SetTrackingSpace(

Valve.VR.ETrackingUniverseOrigin.TrackingUniverseSeated);

}

}

}

**OnCollisionFloor:**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class OnCollisonFloor : MonoBehaviour

{

void Start()

{

GameController.Started = false;

}

void Update()

{

}

void OnCollisionEnter(Collision Floor)

{

if(Floor.gameObject.tag == "TP")

{

GameController.Started = true;

}

}

}

**OnCollisionTP:**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class OnCollisionTP : MonoBehaviour

{

public Rigidbody TP1;

public Rigidbody TP2;

public Rigidbody TP3;

public Rigidbody TP4;

public Rigidbody TP5;

public Rigidbody TP6;

public Rigidbody TP7;

public Rigidbody TP8;

public Rigidbody TP9;

public Rigidbody TP10;

public Rigidbody TP11;

public Rigidbody TP12;

public Rigidbody TP13;

public Rigidbody TP14;

public Rigidbody TP15;

void Start()

{

TP1 = GetComponent<Rigidbody>();

TP2 = GetComponent<Rigidbody>();

TP3 = GetComponent<Rigidbody>();

TP4 = GetComponent<Rigidbody>();

TP5 = GetComponent<Rigidbody>();

TP6 = GetComponent<Rigidbody>();

TP7 = GetComponent<Rigidbody>();

TP8 = GetComponent<Rigidbody>();

TP9 = GetComponent<Rigidbody>();

TP10 = GetComponent<Rigidbody>();

TP11 = GetComponent<Rigidbody>();

TP12 = GetComponent<Rigidbody>();

TP13 = GetComponent<Rigidbody>();

TP14 = GetComponent<Rigidbody>();

TP15 = GetComponent<Rigidbody>();

}

// Update is called once per frame

void Update()

{

}

void OnCollisionEnter(Collision TP)

{

if(TP.gameObject.tag == "Interactable")

{

TP1.useGravity = true;

TP2.useGravity = true;

TP3.useGravity = true;

TP4.useGravity = true;

TP5.useGravity = true;

TP6.useGravity = true;

TP7.useGravity = true;

TP8.useGravity = true;

TP9.useGravity = true;

TP10.useGravity = true;

TP11.useGravity = true;

TP12.useGravity = true;

TP13.useGravity = true;

TP14.useGravity = true;

TP15.useGravity = true;

}

}

}

**Assets:**

Steam VR Assets:

<https://assetstore.unity.com/packages/tools/integration/steamvr-plugin-32647>

VR Interactions:

<https://assetstore.unity.com/packages/tools/input-management/vr-interaction-119934>

Hammer Assets: <https://assetstore.unity.com/packages/3d/props/tools/stylized-hammer-162874>

Spider Assets: <https://assetstore.unity.com/packages/3d/characters/animals/animated-spider-22986>