***Introduction to Graphics and Mobile Gaming***

**LAB 4: Part 2**

**Visuals**

**Issue 1.0**

Contents

[1 Introduction 1](#_Toc36126102)

[2 Visual effects 1](#_Toc36126103)

[3 Dynamic effects 2](#_Toc36126104)

# Introduction

In this lab, we will look at adding more features to our game; this will include visual effects.

# Visual effects

We would like to add some visual flourishes to our game in the form of particle effects.

The first thing to do is to get the particles folder into unity. Go to Assets, create a new folder, and name it ***ParticleSystems***. Now copy everything from the particles folder from the LabResources into your new folder. Even though you will not be using the entire package, download all of the content.

Next, we want to attach the ***ExplosionMobile*** particle effect from the Prefabs folder to any chess piece:

* Make it a child of a chess piece from the hierarchy panel.
* Set force to 0, delete all of the children of explosion mobile except for the ***Sparks*** in the Hierarchy panel.
* Click on Sparks from Hierarchy and move the effect to be within the chess piece in our scene.
* Within the chess piece, Add Component -> New Script from Inspector tab and rename it explosionController. Open the script and add the following line before the start function:

“public Transform sparkle;”

* Save and press Play. Stop the emulation and drag the sparks element from the hierarchy into the Inspector panel of the chess piece and attach it to the new “sparkle” field of the script. In the Start() function of our code, add the following code:

sparkle.GetComponent<ParticleSystem> ().enableEmission = false;

* Click on the Sparks from the hierarchy and then in the Inspector panel, set the duration to 0.50 and tick the Looping option. Save and close. Now, when you play the application again, the particle system should be off automatically.

Finally, we want to delete the update function and instead use the following code:

void OnCollisionEnter()

 {

 sparkle.GetComponent<ParticleSystem>().enableEmission = true;

 StartCoroutine (stopSparkles ());

 }

IEnumerator stopSparkles()

 {

 yield return new WaitForSeconds(.4f);

 sparkle.GetComponent<ParticleSystem>().enableEmission = false;

 }

Save the code and test our scene by colliding two chess pieces; we should see the new particle system working on the chess piece we have attached it to.

You might notice that the Castle sparkles once just as you start the game. This is because Unity detects a collision with either the chessboard plane or the Quad that you have placed on the floor. There’s an easy get around to avoid that, just wrap the start of the animation in an if() statement as below:

void OnCollisionEnter(Collision col)

{

 // Detect which object you want to collide with/don’t want to collide with

 if(col.gameObject.name != "ChessBoard" && col.gameObject.name!= "Quad (5)" ) //Quad(5) is the floor

 {

 sparkle.GetComponent<ParticleSystem>().enableEmission = true;

 StartCoroutine( stopSparkles() );

 }

}

In the case that you are still getting sparkles, you might need to add the skeletonRoom and the plane to the if statement. And don’t forget to check that your pieces are above these gameObjects

void OnCollisionEnter(Collision col)

{

 // Detect which object you want to collide with/don’t want to collide with

 if(col.gameObject.name != "ChessBoard" &&

 col.gameObject.name!= "Quad (5)" && ) //Quad(5) is the floor

 col.gameObject.name!= "roomSkeleton" &&

 col.gameObject.name!= "Plane" )

 {

 sparkle.GetComponent<ParticleSystem>().enableEmission = true;

 StartCoroutine( stopSparkles() );

 }

}

# Dynamic effects

We will now create some dynamic effects to further improve the intractability that the user has with our game.

* Create a new object and rename it ***roomEffects***.
* Create a second new object and rename it ***chimneyFire***, make it a child of roomEffects.
* From the particles folder, go to Prefabs and drag the ***fireMobile*** script to chimneyFire to also make it a child.

Now let’s go to the inspector panel for fireMobile:

* Untick the Particle System Destroyer script
* Click Open Editor and then click on Size over Lifetime in the options and then on the rectangle next to Size. This will bring up a graph that represents the size of the fire particle. Modify it to 15 by clicking on the left-hand side column. See the image below for how this should look



We would like to be able to tap on the fire to turn it on or off:

* Add a Rigidbody and Box collider to the chimneyFire element in the Inspector panel and then rescale the box collider to surround the fire. This is because we want the collider to activate the fire. Adjust the size of the effect to match the interior of the fire place in our scene.
* Change its layer to *particlesEffect*, by adding it as a new layer. Click on room effects from hierarchy and add a new layer called ***particlesEffect*** (make sure your name matches exactly, otherwise the script won’t work). Import the “dynamicParticles” script from this lab folder to the scripts folder (you can find it under *OriginalFiles*; this file will provide a skeleton for your script)

Using the same logic used for setting the particles visible in the chess pieces and applying a force when tapping on them, add the necessary code so that when tapping on a “particleEffect,” the user activates that effect; we want to do this for the fire in the chimney. The user has to be able to switch the particle on and off with every tap.

Firstly, remember that you will need to create a Transform function and attach the script to the fireMobile in the Inspector. You will also need to fill the transform field of the script with fireMobile itself from the hierarchy panel.

void Start()

 {

 layerMask = 1 << LayerMask.NameToLayer("particlesEffect");

 //TODO: Disable the fire in the beginning

 }

 if (e.type == EventType.MouseUp)

 {

 //TODO : fire on and off on click

 hitGO=null;

 }

if ((fingerId == touch.fingerId) && (touch.phase == TouchPhase.Ended || touch.phase ==

 TouchPhase.Canceled))

 {

 //TODO: fire on and off on touch

 hitGO=null;

 }

Add the script to the chimneyFire element, and in the Inspector panel, drag and drop the fireMobile element to the script Transform field. If you get stuck, consult the Solutions folder.

When pressing play, you might notice the fire will slowly fall; this is because the fireplace has no mesh collider. Double-click on the fireplace and add a mesh collider component. Play again and wait; the fire will not move away. Alternatively, just untick the Use Gravity on the chimneyFire.