WENDY ACHIENG MUDENYO

18/02403

BSD

MODE OF STUDY: DAY

MOBILE GAMING PROGRAMMING

QUESTION ONE

1. Discuss Role Of Game Theory In Business
2. In Java game development what does the animation thread class holds
3. Discuss FOUR Android tools used for developing games?
4. Explain TWO advantages and TWO disadvantages of using packed buffers
5. Demonstrate how GUI controls can be added in unity when editing a game

QUESTION TWO

1. Demonstrate how to create a particle system in a scene

First start by opening a new project in Unity. Label the project **Sample28** and include the following assets folders.

* Materials which holds the fire materials
* Scenes which contain the torch scenes
* Textures which contain the texture of the fire material
* Scripts which hold the initial scripts

Open the Torch scene from the Project window and run the scene. Displaying on the screen would be a torch hanging from a wall.

Select TorchFireParticles in the hierarchy. In the Inspector click the Add Component button. Search for Particle System and click on it to add it. Playing the scene would show particles emitting from the torch.

Taking a look at the inspector, it is observed that the particle system has several subsections. Each is called a module which contain settings which describe the particle system

The emission module handle the number and timing of emitted particles in the system to create a continuous flow or even a sudden burst of particles.

You can change the particle system’s shape using the Shape module.

1. Discuss how you can develop the game Chaos ball incorporating; Chaos ball arena, entities and control Objects

First start by opening a new project in Unity. Label the project **Sample32** and include the following assets folders:

* A walled terrain to act as the arena
* Textures for the balls and chaos balls
* Materials which contain the bouncy physics materials
* Interactive scripts which hold the corresponsive scripts

The Arena is created first through the following steps:

Adding a terrain to the project. Set the resolution of the terrain to 50 by 50. Add a directional light to the same scene. Delete the main camera and add a cube to the scene.

Place a cube at (0, 1.5, 25) and scale it to (1.5, 3, 51) …this in return becomes one side of the wall to the arena. Rename the cube to Wall1. Save the scene as Main in the scenes folder

In order to add the texturing, create a folder called Materials. Add a material and name it WallMaterial.

Apply the Cliff(Layered Rock) texture to WallMaterial. Set the x axis tiling to 10. Click and drag the wall material onto the wall1 object in the Scene View.

Next is to texture the ground. Select the terrain and choose the texturing tool in the inspector. Click on Edit Textures and add texture. Select the Grass texture in the dialog box and lick Add. the terrain is now textured with grass.

Next is adding the bouncy material. Right click on the Materials folder and select create the Physics Material and name it BouncyMaterial. Click and drag the Bouncy material onto the wall object in the scene.

Finish the arena by adding the remaining walls. Duplicate Wall1 once and place it at instance (50, 1.5, 25) duplicate the wall again and place it at (25, 1.5, 0) with rotation (0, 90, 0). Duplicate this wall and place it at (25, 1.5, 50).

Game Entities include:

* The player. In this game, the player is a modified First Person character controller.Import the Character controller’s package. Select and drag First Person character controller into the scene. Place the controller at (46, 1, 4) with a rotation of (0, 315, 0) but first move the camera up ad away from the controller.
* Chaos Balls. These are the fast balls flying around the arena. Add them by adding a spere to the scene and rename them Chaos and position them at (15, 2, 25) with a scale of (0.5, 0.5, 0.5)

Click and drag the bouncy material onto the sphere.

Create a material for the chaos balls and select the color the drag the material onto the sphere.

Create a script with the following code and attach it to the ball.

using UnityEngine;

using System.Collections;

public class VelocityScript : MonoBehaviour

{ public float max = 50;

Use this for initialization

void Start () {

rigidbody.velocity = new

Vector3(Random.Range(0,

max),

0,

Random.Range(0,

max));

}

Update is called once per

frame void Update () {

}

}

* The colored balls. Duplicate the ball created four times changing each property to different colors and materials

The control objects include:

* The goals

Each corner has a specific colored goal that corresponds with the colored ball. When the ball enters, the goal checks its tag.

Position the game object at (1.6, 2, 1.6) attach a box collider to the goal and check the IsTriggered property. Change the box collider to be (1.5, 1.5, 1.5)

The following is a script attached to the goal

using UnityEngine;

using System.Collections;

public class GoalScript : MonoBehaviour {

private bool solved = false;

Use this for initialization

void Start () {

}

Update is called once per

frame void Update () {

}

void OnTriggerEnter(Collider other)

{

if(other.tag == tag)

{

solved = true;

other.rigidbody.isKinematic = true;

}

}

public bool IsSolved()

{

return solved;

}

}

Duplicate it to the all the four corner goals. Change the tag to the corresponding colors.

* The Game Controller

Add an empty game object and rename it GameController.

Add a script with the following code:

using UnityEngine;

using System.Collections;

public class GameControlScript : MonoBehaviour {

public

GoalScript

red;

public

GoalScript

blue;

public

GoalScript

orange;

public

GoalScript

green;

private bool isGameOver = false;

// Use this for initialization

void Start () {

}

// Update is called once per frame

void Update () {

if(red.IsSolved() && blue.IsSolved() && orange.IsSolved() &&

green. IsSolved())

{

isGameOver = true;

}

}

void OnGUI()

{

if(isGameOver)

{

GUI.Box(new Rect(Screen.width / 2 - 100,

Screen.height / 2 - 50, 200, 75), "Game Over");

GUI.Label(new Rect(Screen.width / 2 - 30, Screen.height /

2 - 25, 60, 50), "Good Job!");

}

}

}

With the game controller selected, click and drag each goal to its corresponding property to the Game Control script

1. Write a c sharp program that implements prefabs

Objects which are instances of prefabs appear as blue.

To create a prefab, first create a folder under assets in the Project View. Right click on the folder select create then Prefab.

Then fill the prefab with something.

using UnityEngine;

using System.Collections;

public class PrefabGenerator : MonoBehaviour {

//We will store a reference to the target prefab from the

inspector public GameObject prefab;

Use this for initialization

void Start () {

}

Update is called once per

frame void Update () {

//Whenever we hit the B key, we will generate a prefab at

the //position of the original prefab

//Whenever we hit the space key, we will generate a prefab at

the //position of the spawn object that this script is attached

to if(Input.GetKeyDown(KeyCode.B))

Instantiate(prefab);

if(Input.GetKeyDown(KeyCode.Space))

Instantiate(prefab, transform.position, transform.rotation);

}

}

1. Illustrate how to Adding the Soldier to a Scene
2. Explain how to design the game Gauntlet Runner