THE HONG KONG POLYTECHNIC UNIVERSITY
Department of Electrical and Electronic Engineering

EIE3360 Integrated Project
Tutorial 2 – Survival shooter

1. Objective

In this tutorial exercise, you will use Unity to create an isometric 3D survival shooter game.

2. Preparation

- A PC with Microsoft Windows 10
- Unity 2022.3.x
- Unity Hub
- Android SDK
- Microsoft Visual Studio
- Basic knowledge in C# programming, object-oriented programming and 3D graphics

3. Developing a Unity Android Game

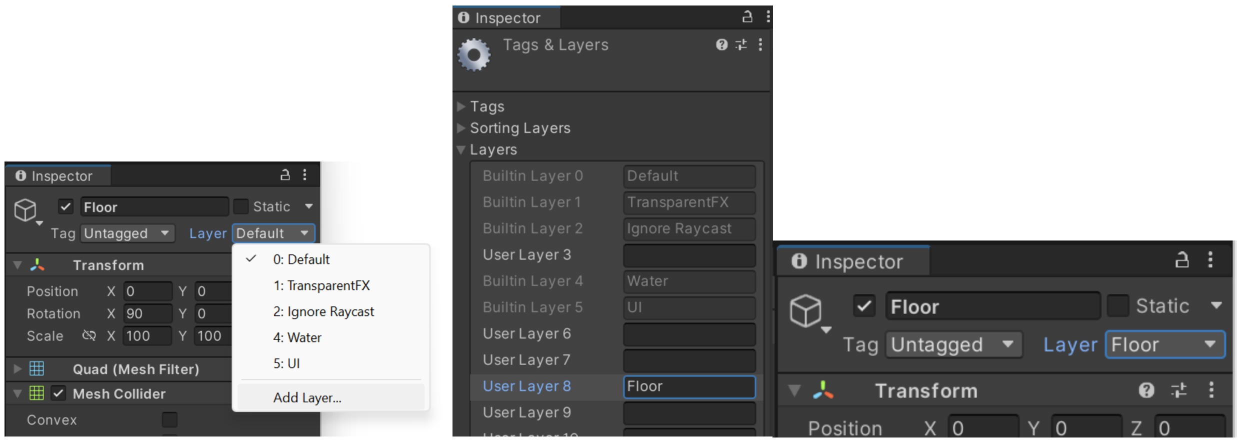
This tutorial exercise uses Unity to create a survival shooter game for Android. In the game, you will create several game objects. The player will control a character to walk around the game board and shoot enemies. You will also detect contact between the character and the enemies.

3.1 Setting up a Unity Game

1. Open and sign-in Unity 2020.3.x with your account.
2. Download the "Survival Shooter" package from Blackboard and create a new **3D project** in Unity
3. In Unity, go to Assets → Import Package → Custom Package, then select the "Survival Shooter" package (select all the components)
4. Delete the Scenes folder. Then, File → New Scene.
5. Save the Scene with the name "Level 01".
6. Click File -> Build Settings → Click "Add Open Scenes" to add the current scene ", Level 01", to the "Scenes In Build" and select the "Level 01" scene to make sure the index is 0.
7. Close the Build Settings and save the project.

1. Create a "Game Scene."

a. Click the Prefabs folder in the Project, and drag and drop "Environment" into the Hierarchy.
b. Ensure the Position is at (0, 0, 0).
c. Drag and drop the "Directional Lights" to the "Environment" in the Hierarchy.
d. Add a plane "Quad" by clicking GameObject -> 3D Object -> Quad and rename it as "Floor". https://gamedev.stackexchange.com/questions/115493/the-better-performance-in-occlusion-culling-plane-or-quad
e. Reset the Position to (0, 0, 0).
f. Set the quad Rotation to X = 90 and the scale = (100, 100, 1).
g. In the Inspector, select the Mesh Renderer. Click the cog ⋮ icon → "Remove Component" to remove it.
h. Set the Layer to "Floor" by adding a layer "Floor" in "User Layer 8" from the Layer dropdown menu.


Inspector
Floor — Static
Tag Untagged — Layer Default
Transform
Position X 0 Y 0
Rotation X 90 Y 0
Scale X 100 Y 100
Quad (Mesh Filter)
Mesh Collider
Convex

0: Default
1: TransparentFX
2: Ignore Raycast
4: Water
5: UI
Add Layer...

Inspector
Tags & Layers
Tags
Sorting Layers
Layers

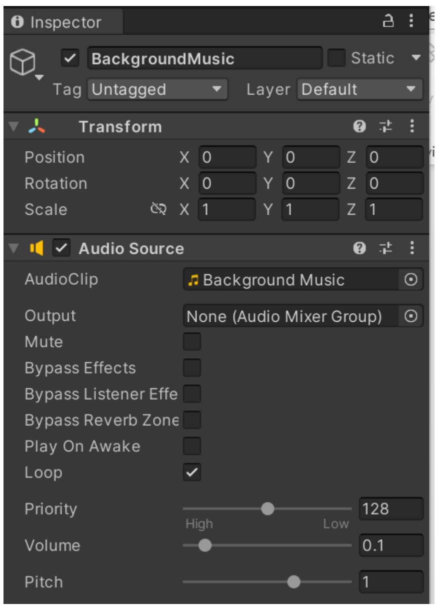
Builtin Layer 0	Default
Builtin Layer 1	TransparentFX
Builtin Layer 2	Ignore Raycast
User Layer 3	
Builtin Layer 4	Water
Builtin Layer 5	UI
User Layer 6	
User Layer 7	
User Layer 8	Floor
User Layer 9	

Inspector
Floor — Static
Tag Untagged — Layer Floor
Transform
Position X 0 Y 0 Z 0

i. File -> Save.

2. Create the “Background Music.”

a. Click GameObject → Create Empty, rename it “BackgroundMusic”.
b. Click Add Component → Audio Source in the Inspector.
c. Click the small icon in the Audio Clip and select “Background Music” clip.
d. Uncheck “Play On Awake”.
e. Check “Loop”.
f. Set the “Volume” to 0.1.


Inspector
BackgroundMusic — Static
Tag Untagged — Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Audio Source
AudioClip: Background Music
Output: None (Audio Mixer Group)
Mute: ☐
Bypass Effects: ☐
Bypass Listener Effe: ☐
Bypass Reverb Zone: ☐
Play On Awake: ☐
Loop: ✓
Priority: 128 (High — Low)
Volume: 0.1
Pitch: 1

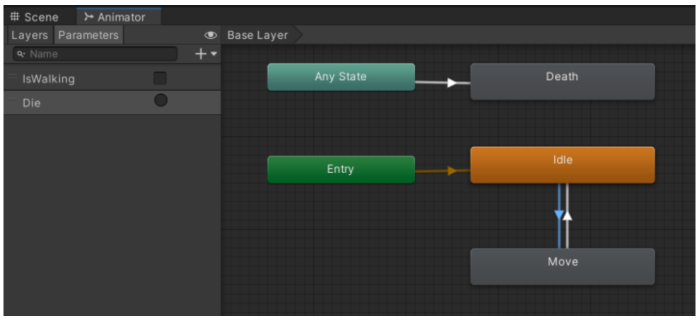
g. File -> Save.

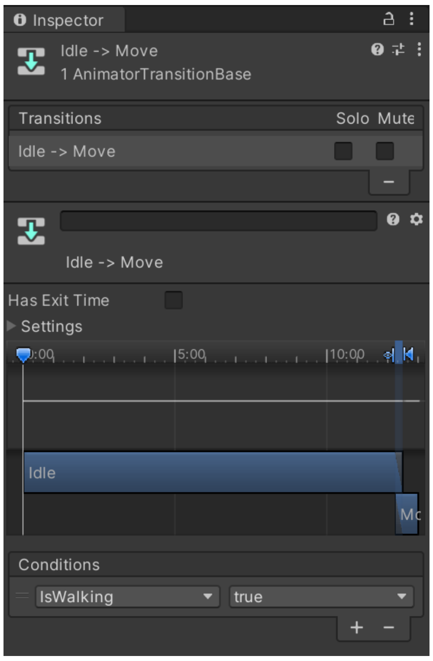
3.2 Player Character

1. Create a “Player.”

a. Go to Assets/Models/Characters in the project windows → drag and drop the “Player” into the Hierarchy.
b. Ensure the Position is at (0, 0, 0).
c. Set the “Tag” to Player.

2. **Create an “Animator” Controller**
   a. Create a folder called “Animation” in Assets, right-click the folder, click Create → Animator Controller, and name it “PlayerAC”.
   b. Drag and drop the “PlayerAC” to the “Player” in the Hierarchy.
   c. Click Window -> Animation -> Animator to open the “Animator” window.
   d. Select “Player” from the Hierarchy and go to Assets/Models/Characters → and expand Player, drag and drop the “Death”, “Idle”, and “Move” motions to the Animator window.
   e. Right-click the “Idle” motion and select “Set As Layer Default State”.
   f. Click a “+” button in the Parameter tab to add a “Bool” parameter and rename it to “IsWalking”, and then create a “Trigger” parameter and rename it to “Die” in the Parameter tab.
   g. Right-click the “Idle” motion → select “Make Transition” and connect it to the “Move” motion with the white arrow.
   h. Select the white line, click the “+” button in the “Conditions” tag and set “IsWalking = true” and uncheck “Has Exit Time”.
   i. Create a transition from “Move” to “Idle” motion and set the condition “IsWalking = false” and uncheck “Has Exit Time”.
   j. Create another transition from “Any State” to “Death” and set the condition to “Die”.




Inspector
Idle -> Move
1 AnimatorTransitionBase
Transitions Solo Mute
Idle -> Move
Idle -> Move
Has Exit Time
Settings
0:00 5:00 10:00
Idle
Conditions
IsWalking true

3. **A character with a physical presence and sound**
   a. Switch to the “Scene” view and select “Player”.
   b. Click “Add Component” → “Rigidbody”.
   c. Set the “Drag” and “Angular Drag” (rotation of a rigid body) to “Infinity” by typing “Inf” and pressing “Enter”. **Drag = tendency of an object to slow down due to friction (i.e., air surrounds an object) https://www.youtube.com/watch?v=86jtkISHars**
   d. Set the Constraints
      - Freeze Position: Y;
      - Freeze Rotation: X, Z.
   e. Click “Add Component” → “Capsule Collider”.
   f. Set the Center = (0.2, 0.6, 0) and the Height = 1.2.
   g. Click “Add Component,” → “Audio Source”, and select “Player Hurt” for the Audio clip.
   h. Uncheck “Play On Awake”.

i. Go to Assets/Scripts/Player, and open the “PlayerMovement” script using MS Visual Studio.
j. Edit the script as below:

```
using UnityEngine;

public class PlayerMovement: MonoBehaviour
{
    public float speed = 6f;    // The speed at which the player will move.

    Vector3 movement;           // The vector stores the direction of the player's movement.
    Animator anim;              // Reference to the animator component.
    Rigidbody playerRigidbody;  // Reference to the player's rigid body.
    int floorMask;          // A layer mask so that a ray can be cast at game objects on the floor layer.
    float camRayLength = 100f;  // The length of the ray from the camera into the scene.

    void Awake()
    {
        // Create a layer mask for the floor layer.
        floorMask = LayerMask.GetMask("Floor");

        // Set up references.
        anim = GetComponent<Animator>();
        playerRigidbody = GetComponent<Rigidbody>();
    }

    void FixedUpdate()
    {
        // Store the input axes.
        float h = Input.GetAxisRaw("Horizontal");
        float v = Input.GetAxisRaw("Vertical");

        // Move the player around the scene.
        Move(h, v);

        // Turn the player to face the mouse cursor.
        Turning();

        // Animate the player.
        Animating(h, v);
    }

    void Move(float h, float v)
    {
        // Set the movement vector based on the axis input.
        movement.Set(h, 0f, v);

        // Normalize the movement vector and make it proportional to the speed per second.
        movement = movement.normalized * speed * Time.deltaTime;

        // Move the player to its current position plus the movement.
        playerRigidbody.MovePosition(transform.position + movement);
    }
    void Turning()
    {
        // Create a ray from the mouse cursor on the screen in the camera’s direction.
        Ray camRay = Camera.main.ScreenPointToRay(Input.mousePosition);

        // Create a RaycastHit variable to store information about what was hit by the ray.
        RaycastHit floorHit;

        // Perform the raycast, and if it hits something on the floor layer...
        if (Physics.Raycast(camRay, out floorHit, camRayLength, floorMask))
        {
            // Create a vector from the player to the point on the floor of the raycast from the mouse hit.
            Vector3 playerToMouse = floorHit.point - transform.position;

            // Ensure the vector is entirely along the floor plane.
            playerToMouse.y = 0f;

              // Create a quaternion (rotation) based on looking down the vector from the player to the mouse.
            Quaternion newRotation = Quaternion.LookRotation(playerToMouse);

            // Set the player's rotation to this new rotation.
            playerRigidbody.MoveRotation(newRotation);
        }
    }

    void Animating(float h, float v)
    {
        // Create a true Boolean if either of the input axes is non-zero.
        bool walking = h != 0f || v != 0f;

        // Tell the animator whether or not the player is walking.
        anim.SetBool("IsWalking", walking);
    }
}
```

k. Save the script.
l. Drag and drop the script “PlayerMovement” to the “Player”.

m. Press the “Play” button to test the game.

n. Save the scene and the project.

Checkpoint 1: Demo to the instructor or teaching assistant

3.3 Camera Setup

1. A script to control the camera to follow the player

a. Set the “Main Camera” position to (1, 15, -22) and the rotation to (30, 0, 0).

b. Set the “Projection” mode to “Orthographic “, the "Size” = 4.5 and the Background to “Black”.

c. Create a folder in Assets/Scripts and name it “Camera”.

d. Create a new C# script called “CameraFollow” in the Assets/Scripts/Camera folder.

e. Drag and drop “CameraFollow” to the “Main Camera".

f. Edit the script as below.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CameraFollow : MonoBehaviour
{
    public Transform target;      // The position that that camera will follow.
    public float smoothing = 5f;  // The speed with which the camera will follow.

    Vector3 offset;               // The initial offset from the target.

    void Start()
    {
        // Calculate the initial offset.
        offset = transform.position - target.position;
    }

    void FixedUpdate()
    {
        // Create a position the camera is aiming for based on the offset from the target.
        Vector3 targetCamPos = target.position + offset;

        // Smoothly interpolate between the camera's current and target positions.
        transform.position = Vector3.Lerp(transform.position, targetCamPos, smoothing * Time.deltaTime);
    }
}
```

g. Save the script.

h. Drag the “Player” object into the “Target” slot in the “Camera Follow (Script)” component in the Inspector.

i. Drag the “Player” into the Prefabs folder in the project window.

j. Enter game mode, and you will see the camera following the player.

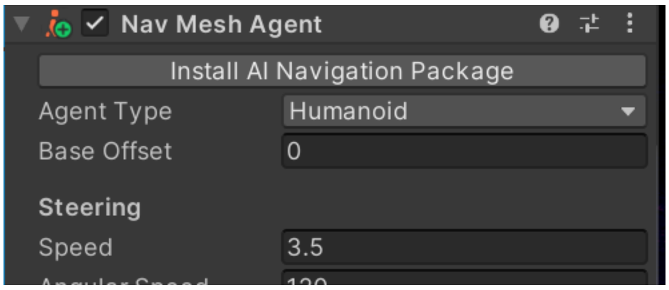
3.4 Creating Enemy

1. Create an enemy to damage the player and follow the player. You need to

- **Add an enemy “Hellephant”**
- **Apply the hit particles**
- **Set the layer**
- **Add the rigid body**
- **Add the capsule collider**
- **Add the sphere collider**
- **Add the audio source**
- **Add the navigation mesh agent**
- **Baking the scene**

a. Go to the Models/Characters folder in the project window, drag and drop the “Hellephant” into the scene near the “Player”.

b. Drag and drop the "HitParticles" from the Prefabs folder onto the parent "Hellephant" in the Hierarchy.
c. Select the "Hellephant" in the Hierarchy and go to the drop-down menu of "Layer", and add a "Shootable" in "User Layer 10" in the inspector. Then, click "Yes" to apply all changes for all child objects.
d. Add a Rigidbody to parent "Hellephant", and set Drag and Angular Drag to "Infinity".
   - Freeze Position Y
   - Freeze Rotation X and Z
e. Add a "Capsule Collider" component, set the Center Y = 0.8 and the Height = 1.5.
f. Add a "Sphere Collider" component, check the "Is Tigger" box, and set the Center Y = 0.8 and the Radius = 0.8.
g. Add an "Audio Source" component, press ⊙ to select "Hellephant Hurt" in the "AudioClip" and uncheck "Play On Awake".
h. Click "Add Component" and type "Nav Mesh Agent" in the search field to add a Navigation Mesh Agent to "Hellephant".
i. Click "Install AI Navigation Package"


Nav Mesh Agent
Install AI Navigation Package
Agent Type: Humanoid
Base Offset: 0
Steering
Speed: 3.5

Note that the "Navigation Mesh Agent" is responsible for moving the characters around a scene and finding paths in a navigation mesh (navmesh).

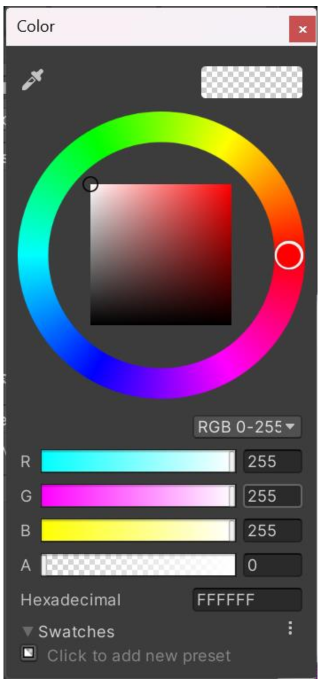
j. Add a GameObject -> AI -> NavMesh Surface to the Hierarchy. Click "Bake" in the Inspector.
k. Go to Window -> AI -> select "Navigation" from the top-up menu. Set the Agent Radius = 0.75, Agent Height = 1.2, and Step Height = 0.38.
   Note that the NavMesh Baking is the process that collects the Render Meshes and Terrains of all Game Objects, which are marked as Navigation Static, and then processes them to create a navigation mesh that approximates the walkable surfaces of the level.

2. Apply the "Animation" to the Enemy
a. Go to the "Assets/Animation" folder, and create an Animator Controller called "EnemyAC".
b. Select parent "Hellephant" in the Hierarchy and drag and drop the "EnemyAC" to the "Hellephant".
c. Double-click "EnemeyAC" to open the "Animator" window.
d. Expand the "Hellephant" model from Models/Character/Hellephant in the project window, drag and drop "Move", "Idle", and "Death" motion into the "Animator" window.
e. Set "Move" motion as "Layer Default State" (by right-clicking the "Move" motion and selecting "Set as Layer Default State")
f. Add a "Trigger" parameter and the name "PlayerDead" and add one more "Trigger" parameter called "Dead".
g. Create a transition from "Move" to "Idle" motion and set the condition to "PlayerDead".
h. Create a transition from "Any State" to "Death" motion and set the condition to "Dead".
i. Go to the Assets/Scripts/Enemy folder, drag and drop the script "EnemyMovement" onto the parent "Hellephant".
j. Enter the game mode and test it.

3.5 Player Health

1. UI system to show game information on the screen
1. Switch to the 2D mode.
2. Create a "Canvas" game object by clicking GameObject → UI → Canvas and rename it to "HUDCanvas".
3. Double-click to "focus" on it.

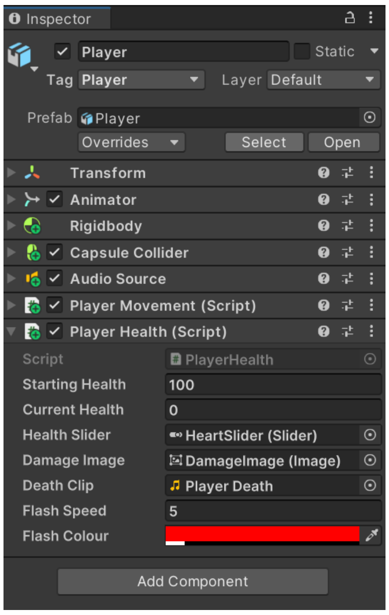
4. Add a "Canvas Group" component by clicking Add Component → Canvas Group.
5. Uncheck the "Interactable" and the "Blocks Raycasts".
6. Right-click the "HUDCanvas" → Create an "Empty" game object and rename it to "HealthUI". Set the "Anchor" to "bottom-left", Pivot = (0, 0), Position = (0, 0, 0), Width = 75 and Height = 60.
7. Right-click the "HealthUI" → UI → Image, rename it to "Heart", and set the Position = (0, 0, 0), Width = 30 and Height = 30.
8. Choose the "Heart" image in the "Source Image".
9. Right-click the "HealthUI" → UI → Slider and rename it to "HealthSlider" and set the Position = (95, 0, 0).
10. Expand the "HealthSlider", select the "Handle Slide Area", and delete it.
11. Select the "HealthSlider" game object again, and set the "Transition" to "None" and "Max Value" to 100, "Value" = 100.
12. Expand the "HealthSlider" game object , select the "Fill Area" game object, and set Right = 0.
13. Select the "HealthUI" game object and add another UI → Image called "DamageImage", drag up and drop it onto the "HUDCanvas" game object and make it a child of "HUDCanvas" instead.
14. Set the "DamageImage" anchors to stretch-stretch, Left = 0, Right = 0, Top = 0, Bottom = 0, Pos Z = 0 and the Alpha Color to 0.


Color

RGB 0-255

R 255
G 255
B 255
A 0
Hexadecimal FFFFFF

2. **Implement the player health**
   a. Go to the Assets/Scripts/Player folder, drag and drop the script "PlayerHealth" onto the "Player".
   b. Select the "Player" in the Hierarchy and drag and drop the "HealthSlider" to the "Health Slider" properties of the "Player Health (Script)" component.
   c. Select the "Player" again in the Hierarchy and drag and drop the "DamageImage" to the "Damage Image" properties of the "Player Health (Script)" component.
   d. Set the "Death Clip" to "Player Death" clip.


Inspector

Player — Static

Tag Player — Layer Default

Prefab Player

Overrides — Select — Open

Transform

Animator

Rigidbody

Capsule Collider

Audio Source

Player Movement (Script)

Player Health (Script)

Script	PlayerHealth
Starting Health	100
Current Health	0
Health Slider	HeartSlider (Slider)
Damage Image	DamageImage (Image)
Death Clip	Player Death
Flash Speed	5
Flash Colour	🟥

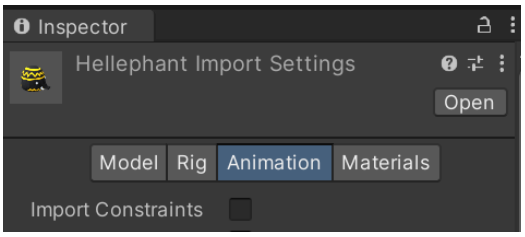
Add Component

e. Go to the Assets/Scripts/Enemy folder, and drag the script “EnemyAttack” onto the parent “Hellephant”.
f. Save the scene and the project.
g. Enter the game mode and test it.

3.6 Harming Enemies

1. Implement the enemy health

a. Go to the Assets/Scripts/Enemy folder, and drag the script “EnemyHealth” onto the “Hellephant”.
b. Set the “Death Clip” to “Hellephant Death”.
c. Go to the Assets/Models/Character folder, select the “Hellephant”, and go to the “Animations” tab in the Inspector.


Inspector

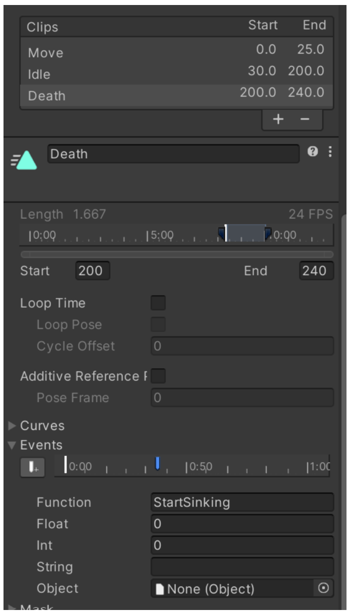
Hellephant Import Settings

Open

Model | Rig | Animation | Materials

Import Constraints

d. Select the “Death” Clips, expand the “Events”, and click the ending time symbol and this function is called “StartSinking” (it will call when an enemy dies).


Clips	Start	End
Move	0.0	25.0
Idle	30.0	200.0
Death	200.0	240.0

Death

Length 1.667 — 24 FPS

Start 200 — End 240

Loop Time	
Loop Pose	
Cycle Offset	0
Additive Reference Pose	
Pose Frame	0

Curves

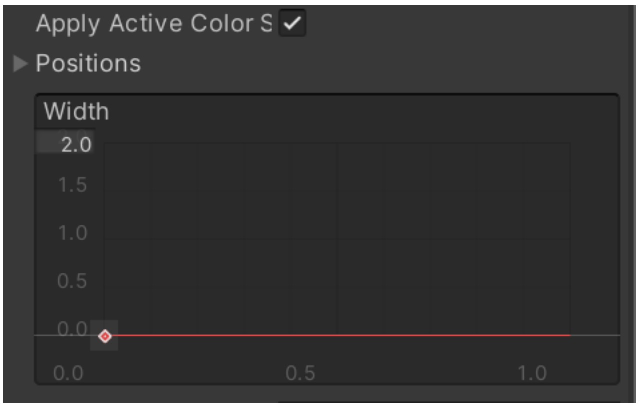
Events

Function	StartSinking
Float	0
Int	0
String	
Object	None (Object)

e. Modify the “EnemyAttack” script, and **uncomment the block scripts**.
f. Modify the “EnemyMovement” script, and **uncomment the block scripts**.

2. Create a gun light (Copy and paste the “GunParticles” from Prefabs to Player).

a. Go to the Assets/Prefabs folder, and select the “GunParticles”. Click the “Particle System” cog icon” and select the “Copy Component”.
b. Go to the “Player” → “GunBarrelEnd” in the Hierarchy, click the cog icon ⋮ and select “Paste”->“ Component As New”.
c. Click the “Add Component → “Line Renderer” to add a line renderer.
d. Expand the “Materials”, and set the “Element 0” to “LineRenderMaterial”.
e. Drag the red dot from the left-hand corner to 0.0 and set the width value to 2.



a. Disable the “Line Renderer” component by unchecking the checkbox next to the “Line Renderer”.
b. Click the “Add Component” → Light and set the Color to “red”.
c. Disable the “Light” component.
d. Add an “Audio Source” component, select “Player GunShot” clip and uncheck “Play On Awake”.
e. Go to the Assets/Scripts/Player folder, and drag and drop the script “PlayerShooting” onto the “GunBarrelEnd” game object.
f. Modify the “PlayerHealth” script, and **uncomment the lines with PlayerShooting**.
g. File -> Save.
h. Enter the game mode and test.

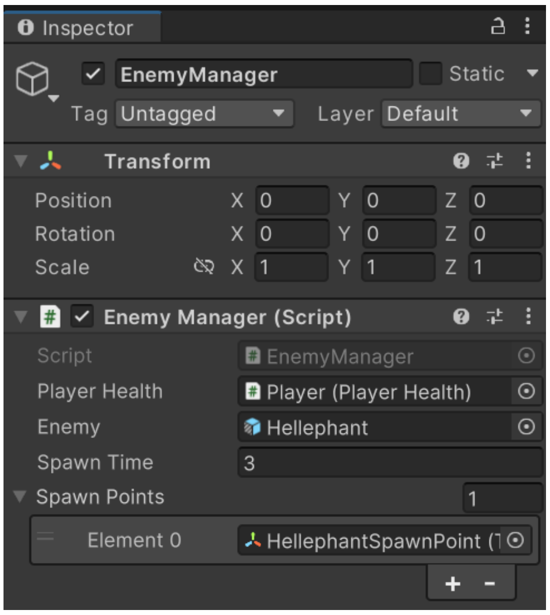
3.7 Scoring Points

1. Add a scoring point system

a. Switch to 2D view.
b. Select the “HUDCanvas” and create a “Text” by right-clicking the “HUDCanvas” game object → select the “UI” → “Legacy” → “Text” and rename it as “ScoreText”.
c. Set the “Anchor” to “top-centre”, Position = (0, -55, 0), Width = 300, Height = 50, Color to “white”, Font to “LuckiestGuy”, Font Size to 30, Text to “Score: 0”, and center alignment.
d. Click the “Add Component” → “Shadow” and set the “Effect Distance” to (2, -2).
e. Go to the Assets/Scripts/Managers folder, drag and drop the script “ScoreManager” to the “ScoreText” in the Hierarchy.
f. Select the “Hellephant” in the Hierarchy, open the “EnemyHealth” script and **uncomment the line**
   - **ScoreManager.score += scoreValue;**
g. Save the scene and the project.
h. Enter the game mode and test.
i. Save the “Hellephant” as a prefab in an Assets/Prefabs folder.

3.8 Spawning Enemies

1. **Create an enemy manager to manage enemy spawning**
   a. Disable the “Hellephant” in the Hierarchy.
   b. Create an empty “GameObject” called “EnemyManager” and reset the position to (0, 0, 0).
   c. Go to the Assets/Scripts/Managers folder, and drag the script “EnemyManager” onto the “EnemyManager” in the Hierarchy.
2. **Set up the spawn point for the enemy**
   a. Create an empty GameObject called “HellephantSpawnPoint”.
   b. Click the cube at the top of the Inspector and select the “yellow” icon.
   c. Set the Position = (-20.5, 0, 12.5) and Rotation = (0, 130, 0)
   d. Select the “EnemyManager” in the Hierarchy, drag and drop “Player” to the “Player Health” field of the “Enemy Manager (Script)”.
   e. Go to the Assets/Prefabs folder, drag and drop the “Hellephant” to the “Enemy” field of the “Enemy Manager (Script)”.
   f. Expand the “Spawn Points” in the “Enemy Manager (Script)” in the inspector and drag and drop “HellephantSpawnPoint” onto the name of Spawn Points.


Inspector
EnemyManager
Static
Tag Untagged
Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Enemy Manager (Script)
Script EnemyManager
Player Health Player (Player Health)
Enemy Hellephant
Spawn Time 3
Spawn Points 1
Element 0 HellephantSpawnPoint (T

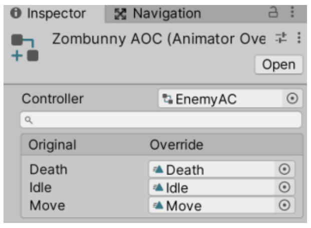
   g. Save the scene and the project.
   h. Enter the game mode and test it.

Checkpoint 2: Demo to the instructor or teaching assistant

4. Exercise 1

In this exercise, you need to add two enemy types. They are “Zombunny” and “Zombear” by using the same techniques to overwrite the “Hellephant” animator controller.

1. Go to the Assets/Animation folder and create an “Animator Override Controller” by right-clicking the empty space and renaming it as “ZombunnyAOC”.
2. Drag and drop the “EnemyAC” onto the “Controller” field of the ZombunnyAOC in the Inspector.
3. Go to the Assets/Models/Characters folder, expand the “Zombunny”, and drag “Move” onto the “Move” slot, “Idle” onto the “Idle” slot, and “Death” onto the “Death” slot.



Zombunny AOC (Animator Ove

Controller	EnemyAC

Original	Override
Death	Death
Idle	Idle
Move	Move

4. Drag the "Zombunny" from Assets/Models/Characters to the scene and apply the "ZombunnyAOC" to Controller in the Animator component.
5. Drag the "Zombunny" to Assets/Prefabs.
6. Drag another copy of the "EnemyManager" to the "EnemyManager" and assign "Zombunny" to the new one.

Please follow the animation steps of Hellephant to create Zombear animation

5. Exercise 2

Modify the game so that

1. There is a UI Text to show your name.
2. Create a "Game Over" trigger after the player's death.
3. Add two enemy types, "Zombunny" and "Zombear", and save them in the prefabs.
4. The first "Zombunny" appears at Position (0, 0, 18) and Rotation (0, 250, 0) every 5 seconds.
5. The first "Zombear" appears at Position = (22.5, 0, -6) and Rotation = (0, 240, 0) every 5 seconds.
6. The score of the enemies:
   a. Zombunny – 15 scores
   b. Zombear – 5 scores
   c. Hellephant – 20 scores
7. Create a WebGL application and upload it to a webserver.

Checkpoint 3: Demo to the instructor or teaching assistant Blackboard

References

[1] Unity - Game Engine, http://unity3d.com/

[2] Unity - Survival Shooter tutorial, https://learn.unity.com/project/survival-shooter-tutorial

[3] Unity Manual, http://docs.unity3d.com/Manual/index.html

[4] Unity - Script API, http://docs.unity3d.com/ScriptReference/index.html