PURPLE BELT SENSEI GUIDE

CODE NINJAS®

Setup and Installation 9
Downloading and Installing Unity Hub 10
Logging Into Unity Hub 15
Installing Unity 2022.3 (LTS) 18
Removing Old Versions of Unity 26
Creating a New Unity Project 27
Opening an Existing Project 32
GitHub 34
Creating a New GitHub Repository 35
Syncing Local Changes to GitHub 39
Syncing an Existing GitHub Repository 41
Syncing Changes from Another Computer 44
Activity Solution: Dropping Bombs 45
Hierarchy 45
Cube Object 46
Bomb Object 47
Activity Solution: Color Drop Prove Yourself 48
Hierarchy 48
Cube Material 49
Sphere Material 50
Cube Object 51
Sphere Object 52
Activity Solution: Scavenger Hunt 53
Hierarchy 53
Avatar Prefab Object 54
Background Object 57
CM vcam1 Object 58
gameicons.png Asset 59
Game Icons Objects 60
Quad objects 61
ScoreSystem Object 62
Text object 63
Activity Solution: Particle Hunt Prove Yourself 64
Hierarchy 64
Game Icons Objects 65

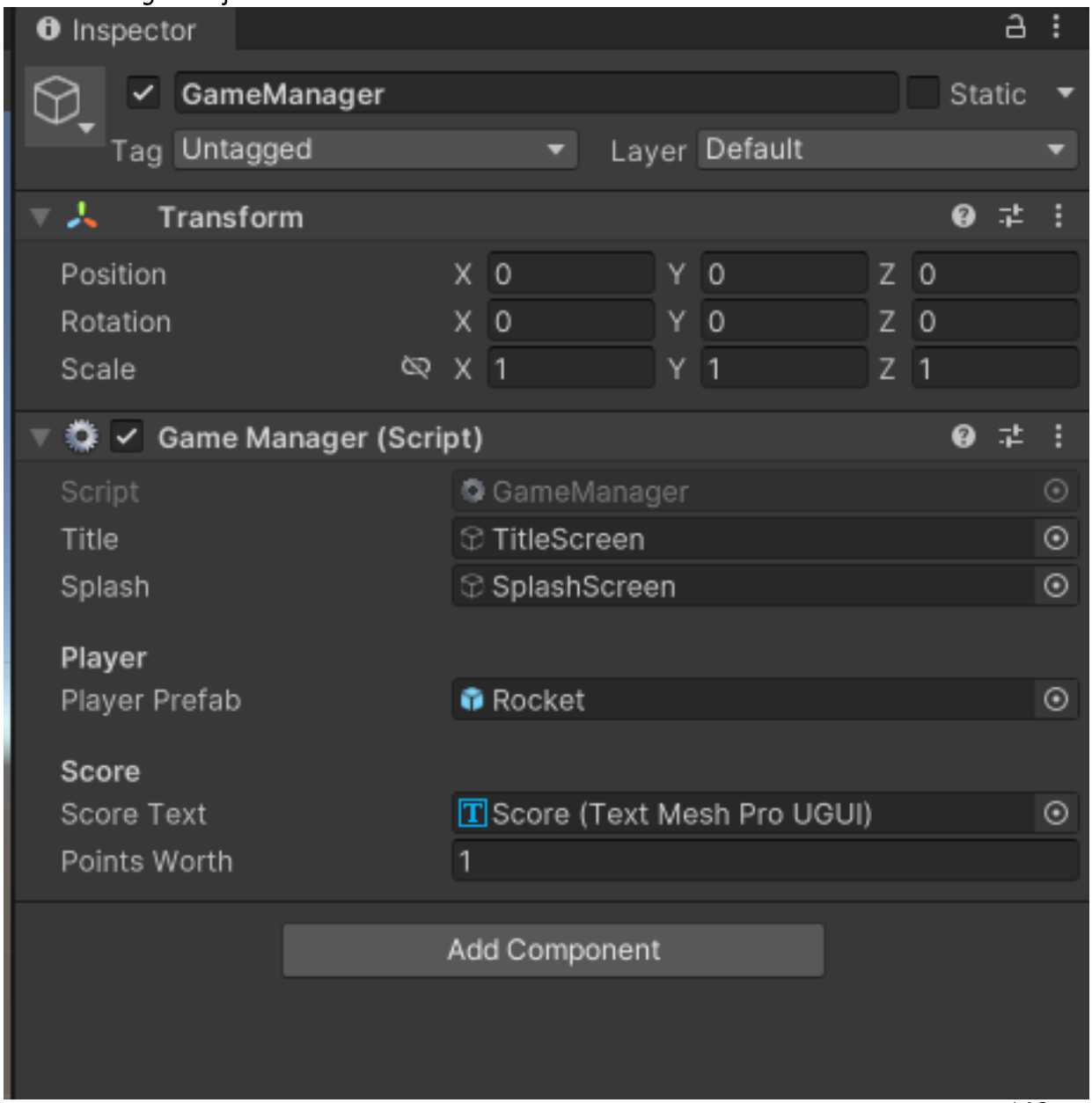
Gem Objects 66
Activity Solution: Meany Bird 67
Hierarchy 67
Background Object 68
Bird Object 69
Collectibles.cs Script 70
DestroyAfterTime.cs Script 70
GameController Object 71
GameController.cs Script 71
Move.cs Script 72
PlayerControls.cs Script 73
Spawner Object 74
Spawner.cs Script 74
Spikes Prefab 75
Spikes > Score Prefab 75
Activity Solution: Meaner Bird Prove Yourself 76
Hierarchy 76
CaveBackground Object 76
CaveBackground > caveback1 Object 77
CaveBackground > caveback2 Object 78
Spikes Prefab 78
Activity Solution: Sketch Head 79
Hierarchy 79
Destroyer.cs Script 79
DoodleHead Object 80
CameraFollow.cs Script 82
GameController Object 82
GameController.cs Script 83
Main Camera Object 84
Main Camera > Destroyer Object 85
Platform Object 86
PlayerControls.cs Script 87
Activity Solution: Trick Head Prove Yourself 88
Hierarchy 88
FakeGameController Object 88

FakePlatform Object 89
Activity Solution: Don't Touch the Cubes 90
Hierarchy 90
GameControls.cs Script 90
GameManager Object 91
Main Camera Object 92
Spawner Object 94
SpawnObjects.cs Script 95
Activity Solution: Don't Touch the Chopsticks Prove Yourself 96
Hierarchy 96
Chopstick Prefab Object 96
Spawner 1 Object 97
Spawner 2 Object 97
Activity Solution: Super Shapes 98
Hierarchy 98
GameController Object 99
GameController.cs Script 100
Main Camera Object 101
Player Object 102
PlayerControls.cs Script 103
Shape Objects 104
Shape.cs Script 105
Rotator.cs Script 106
Activity Solution: Super Duper Shapes Prove Yourself 107
Double Hexagon Prefab Object 107
Double Pentagon Prefab Object 107
Double Triangle Prefab Object 107
GameController Object 108
Six Sides Prefab Object 109
Activity Solution: Polyrun 110
Hierarchy 110
ChallengeObj Prefab Object 111
DestroyAfterTime.cs Script 112
GameController Object 112
GameController.cs Script 113

Move.cs Script 113
PlayerControls.cs Script 114
Player Prefab Object 116
Spawner Object 118
Spawner.cs Script 119
Activity Solution: Polyrun v2 Prove Yourself 120
ChallengeObj Prefab Object 120
Spawner 120
Activity Solution: Dropping Bombs Part 2 121
Hierarchy 121
Bomb Prefab Object 122
Cloud Animation Frames 124
Cloud Object 125
Fly Left Animation Frames 126
Fly Right Animation Frames 126
Main Camera Object 127
RocketAnimate.cs Script 128
Rocket Animation Frames 128
Rocket Animator Tree 129
Rocket Blend Tree 130
Rocket Prefab Object 130
Sky Object 133
Activity Solution: Dropping Bombs Part 3 134
Hierarchy 134
Background Object 135
Bomb Prefab Object 136
GameManager Object 137
GameManager.cs Script 138
GameOver Object 140
PlayAgain Object 141
Rocket Prefab Object 142
Score Object 145
Spawner Object 146
Spawner.cs Script 147
SplashScreen Object 148

TitleScreen Object 149
Press Any Key Text Object 150
Title Object 151
Activity Solution: Dropping Bombs Part 4 152
Hierarchy 152
ExplosionClear.cs Script 153
Explosion Prefab Object 154
Explosion Prefab > Smoke Object 157
Explosion Prefab > Smoke > Debris Object 160

GameManager Object



..... 163

GameManager.cs Script 164

Rocket Prefab Object 166

Rocket Prefab > Particle System 167

Rocket Prefab > Particle System > Particle System..... 170

TriggerExplosion.cs Script.....173

Activity Solution: Dropping Bombs Part 5 174

Hierarchy 174

BestScore Object 175

GameManager Object 176
GameManager.cs Script 177

Setup and Installation

There are three main components of a **Unity** installation.

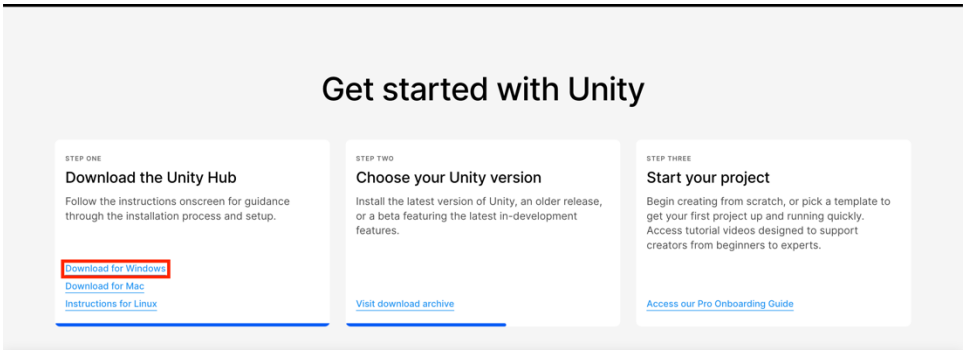
- The **Unity Hub** manages projects and installations.
- The **Unity game engine** is used to create games.
- **Visual Studio Community 2022** is used to code scripts.

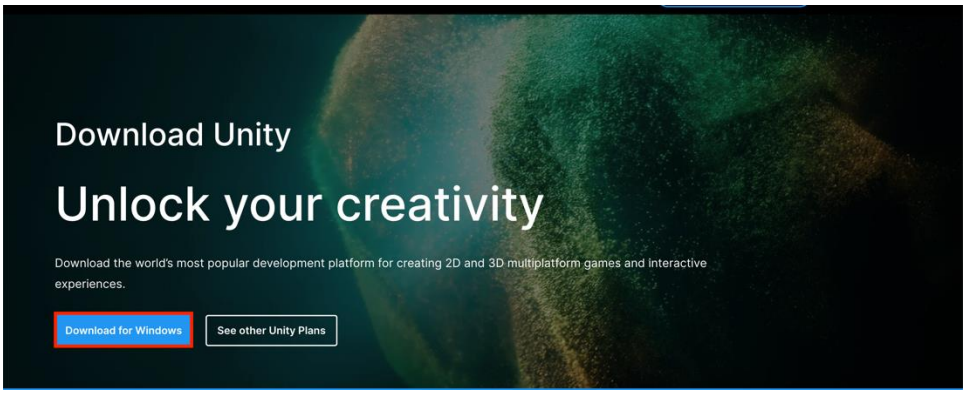
Individual ninja accounts for **Unity** and **Visual Studio** are not required for Ninjas. All project files are stored locally on the computer. You can use a USB-drive to store projects and transfer from computer to computer. Unity Collab, Unity's built in version control system, is not a free service.

You can use GitHub and GitHub Desktop to cloud sync unlimited projects for free. Dropbox or Google Drive are easier to use, but they do not efficiently handle syncing Unity projects because each contains thousands of files.

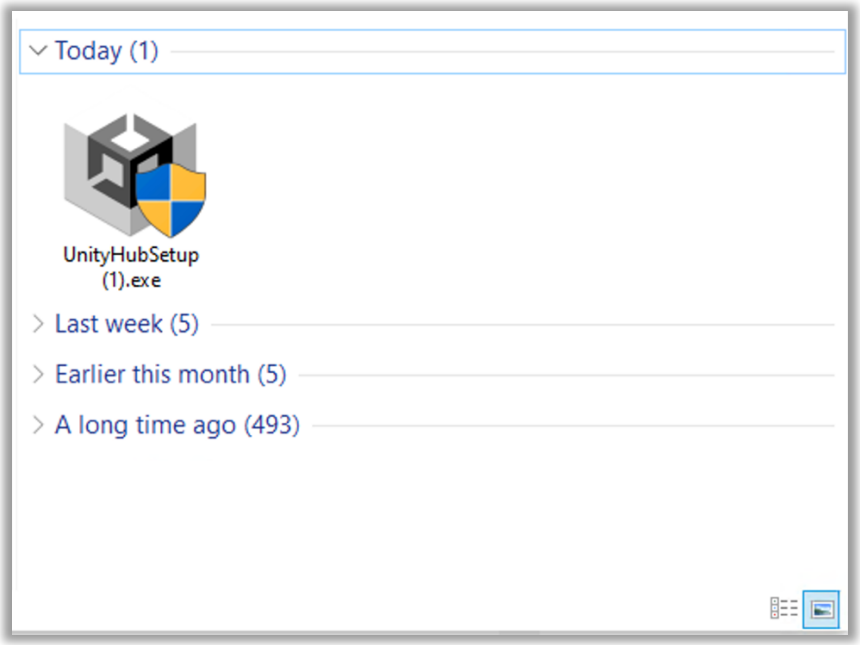
Downloading and Installing Unity Hub

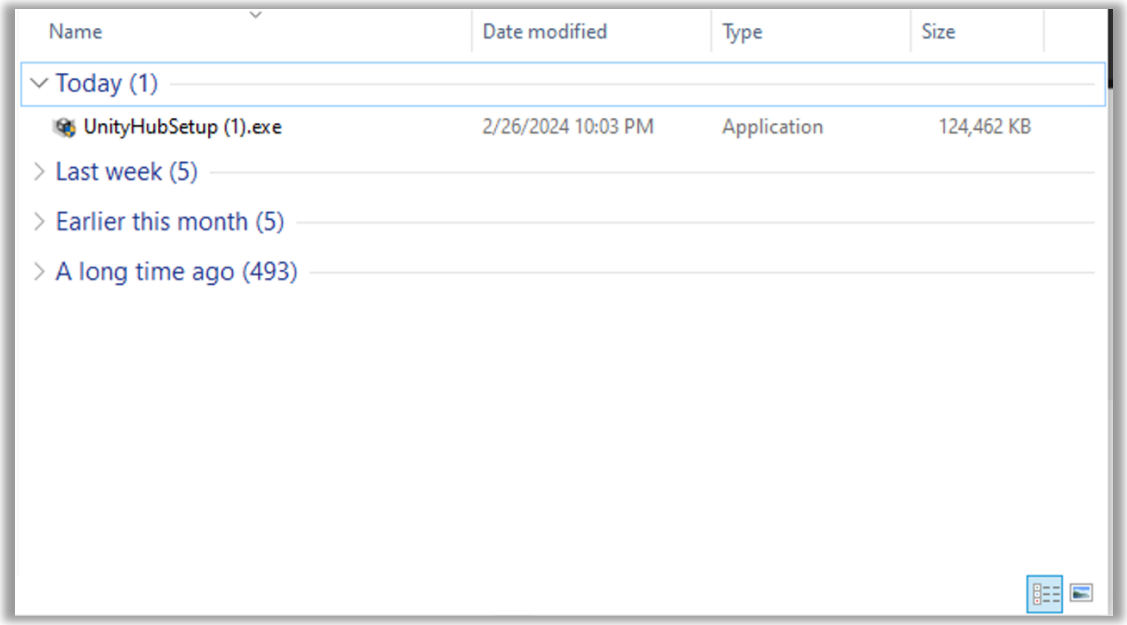
1. Download **Unity Hub**. Do not download the beta version. https://unity3d.com/get-unity/download



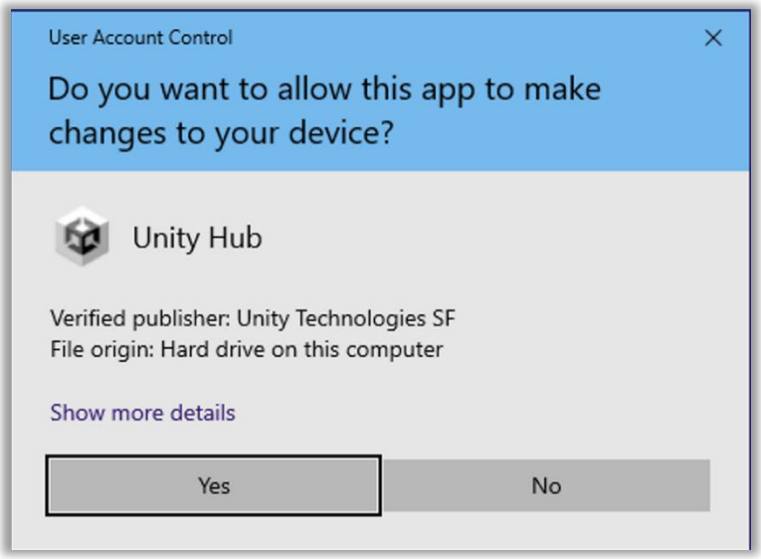


2. After the file downloads, find and run the executable by double clicking on the application.

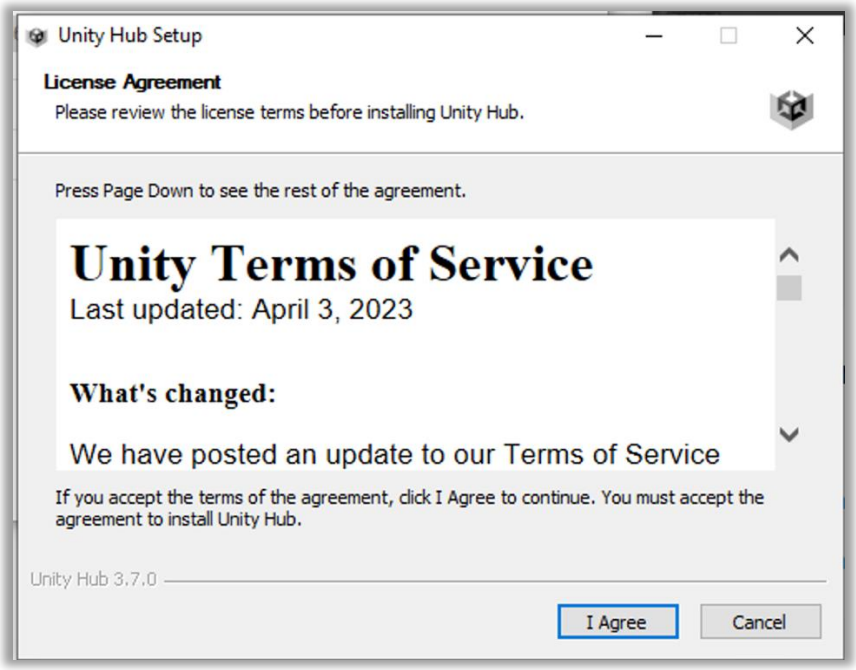




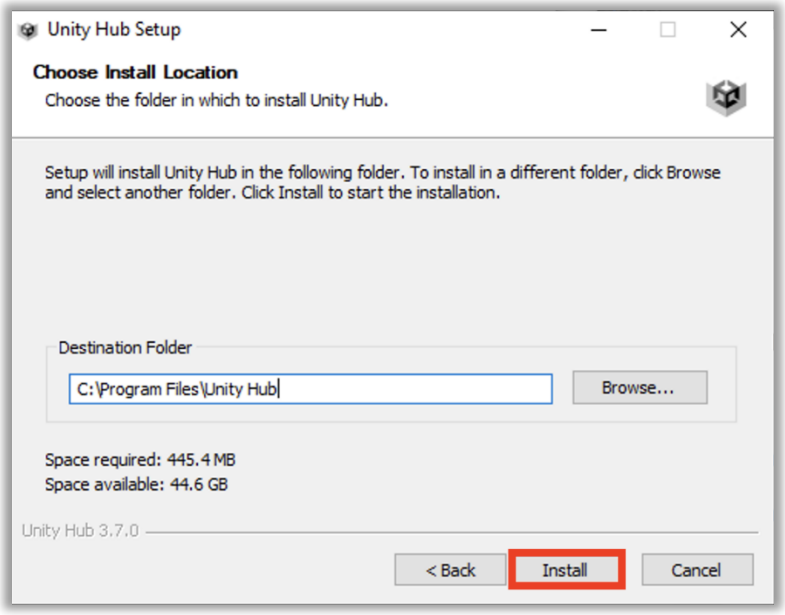
3. If the **Windows User Account Control** opens, click **Yes**.



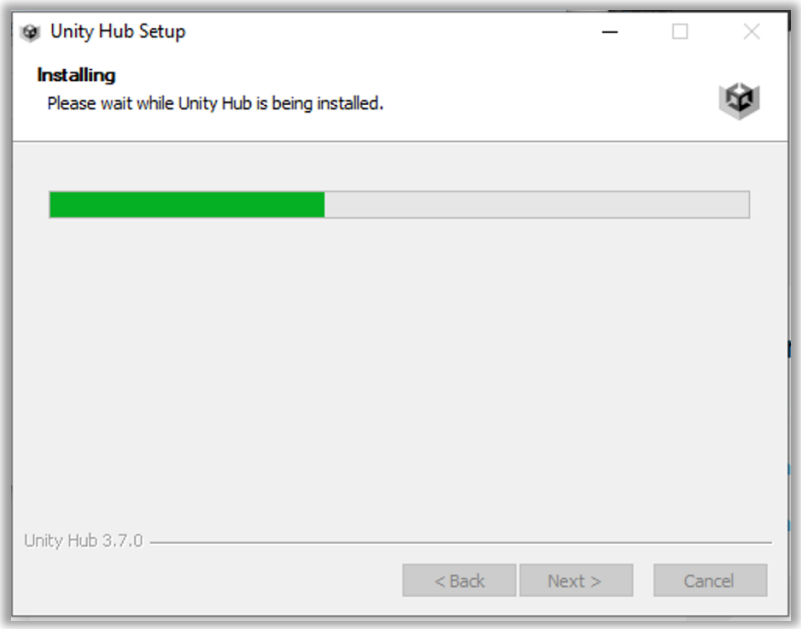
4. Review the License Agreement and click **I Agree**.



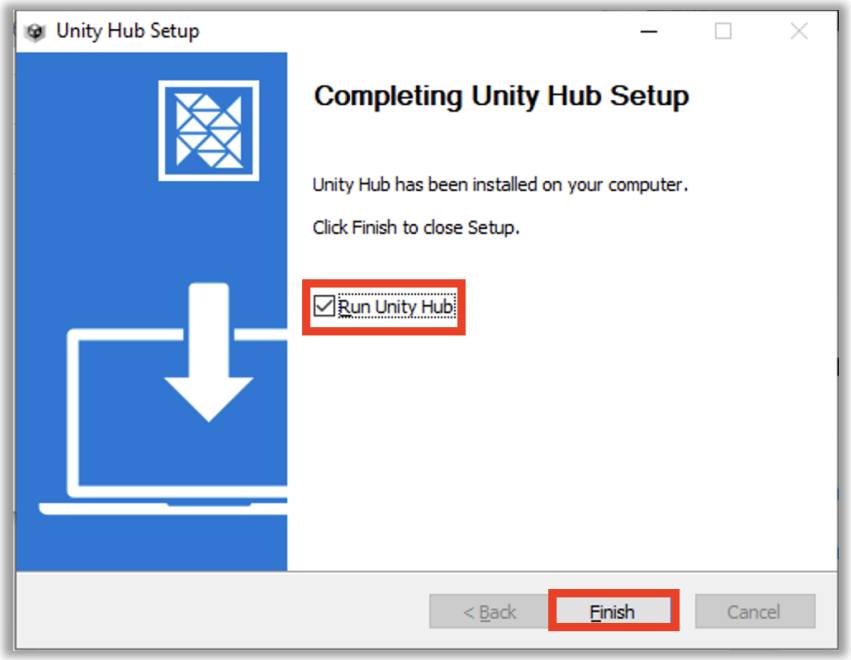
5. Keep the default destination folder and click **Install**.



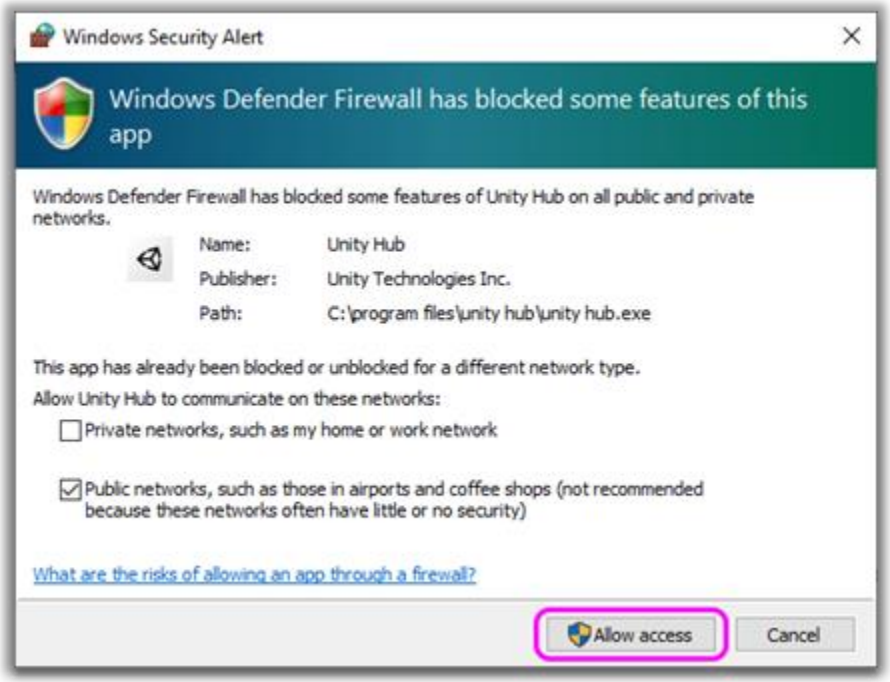
6. Wait for the installer to finish.



7. Check the box to run **Unity Hub** and click **Finish**.

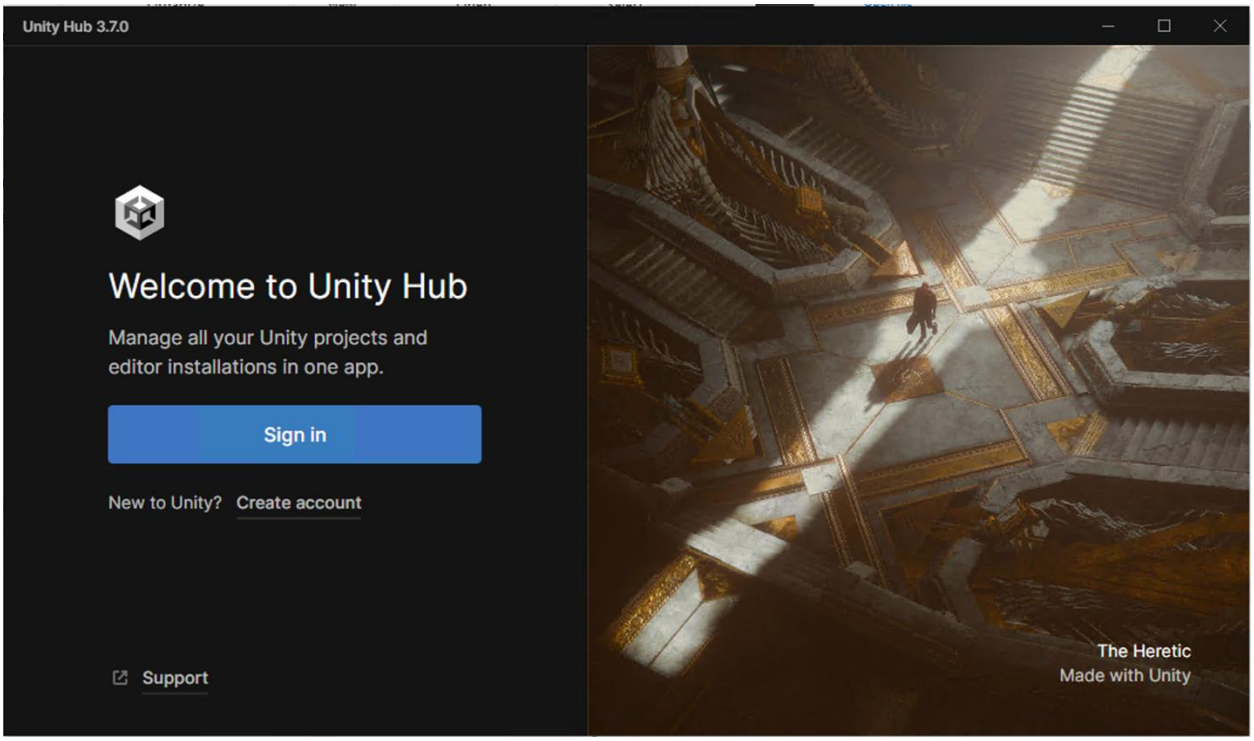


8. If **Windows Security Alert** opens, click **Allow access**.

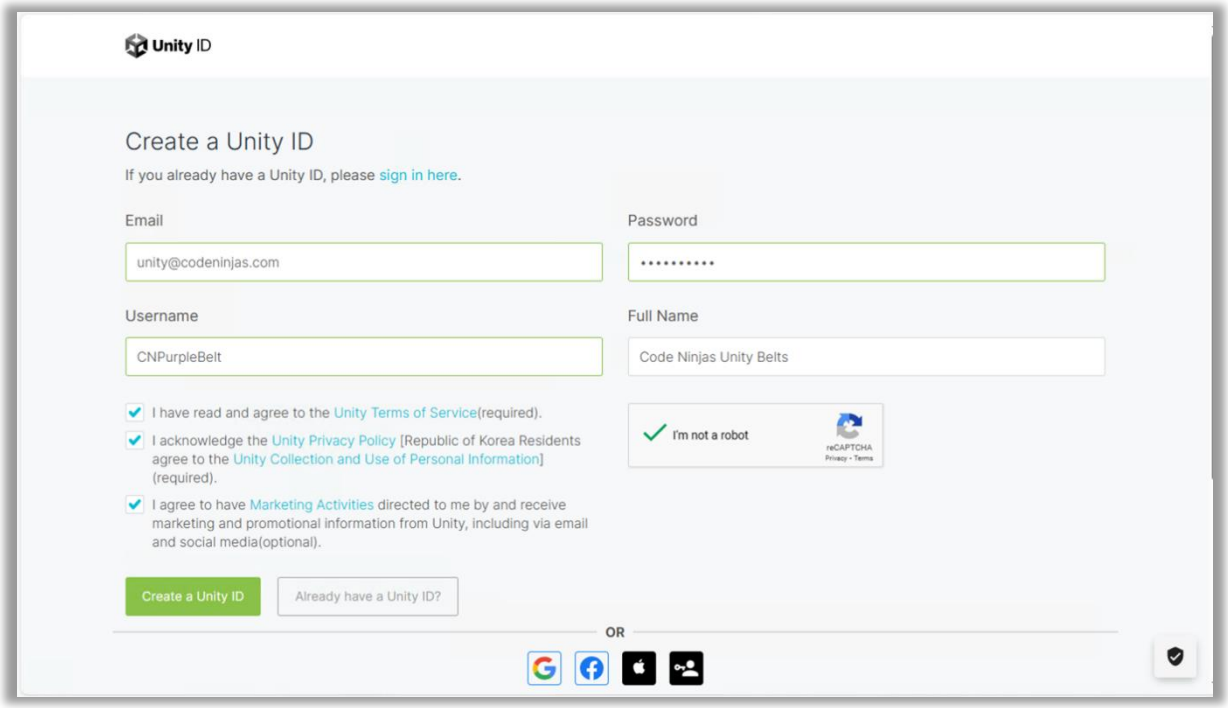


Logging Into Unity Hub

Unity Hub requires a login to proceed to the screen where we can create games and look at versions. If your location already has a process for Unity accounts, you can select the sign in button. Otherwise, to learn about creating accounts, follow along below.

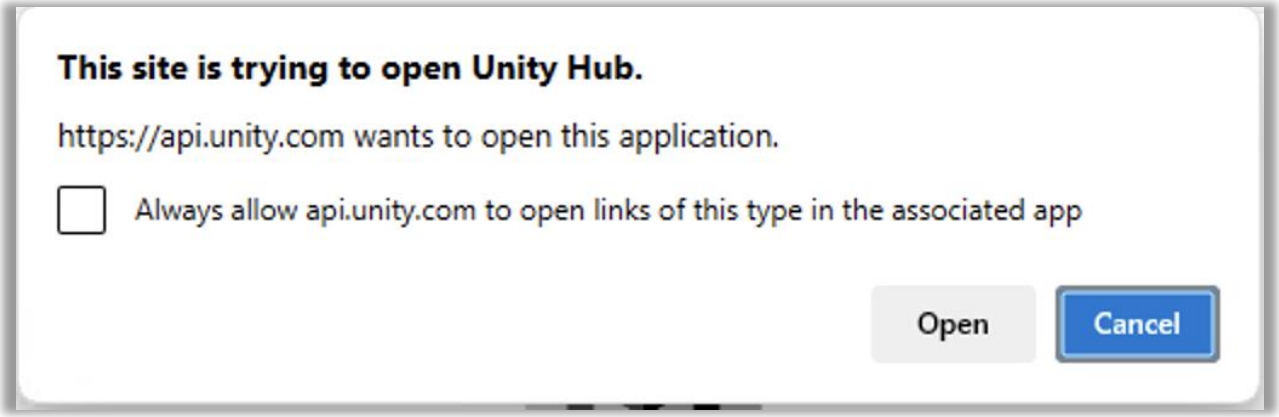


1. Click on **Create Account**. It will open the browser on your computer you have selected as default. This is the screen where you **Create a Unity ID**. You can either choose to fill in the details or you can log on using a Single sign-on (SSO) provider such as Google or AppleID.



2. Once you click **Create a Unity ID**, Unity will send you an email confirmation. Open it and confirm your email then return to the tab and click continue.

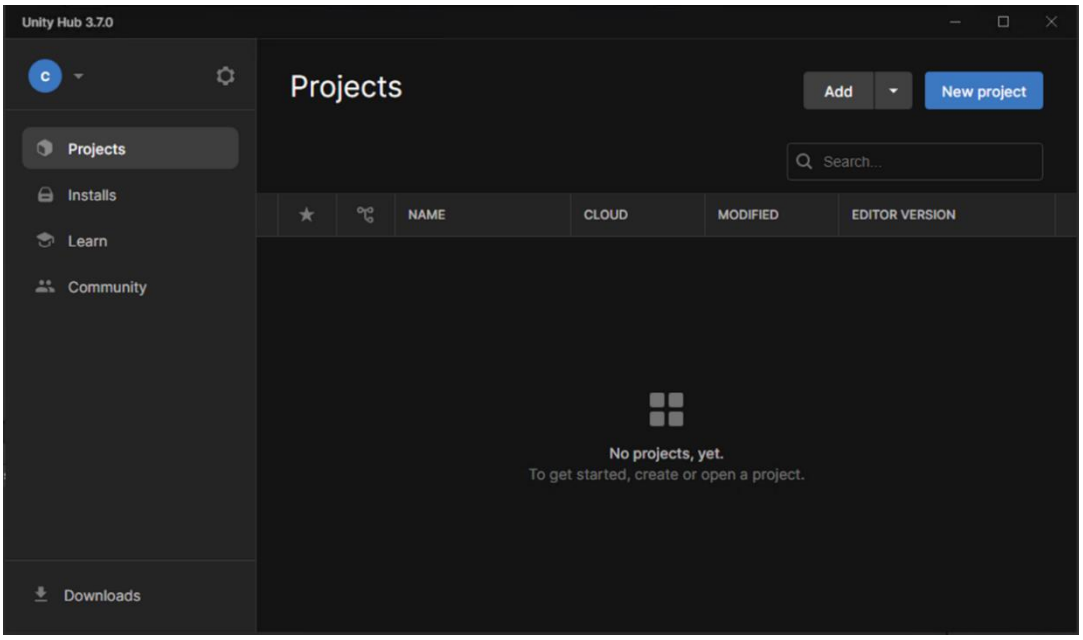
   A message like the one below may appear depending on whether Unity was used in your location before on this specific device. The appearance of the message may differ depending on the browser used.



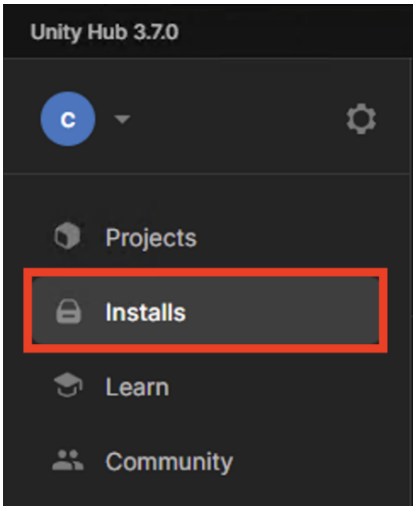
Installing Unity 2022.3 (LTS)

This is the same version of Unity utilized in all of Code Ninjas Unity belts.

1. Once **Unity Hub**, is signed in it will look like the below image.

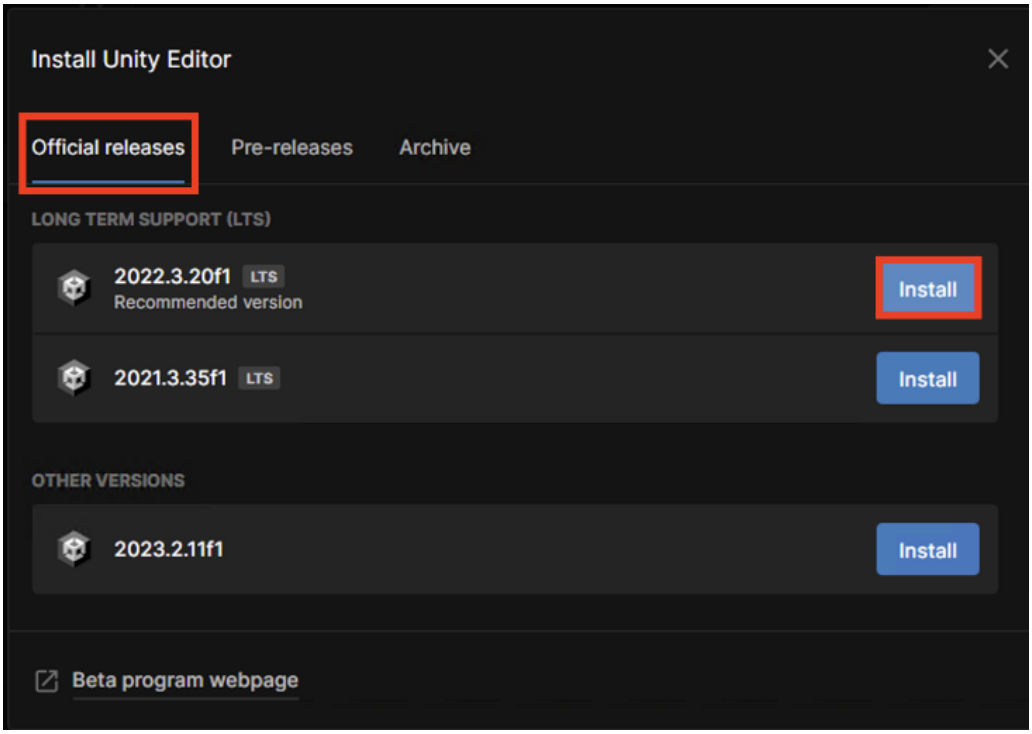


To install the version of Unity we need select the Install tabs on the left.

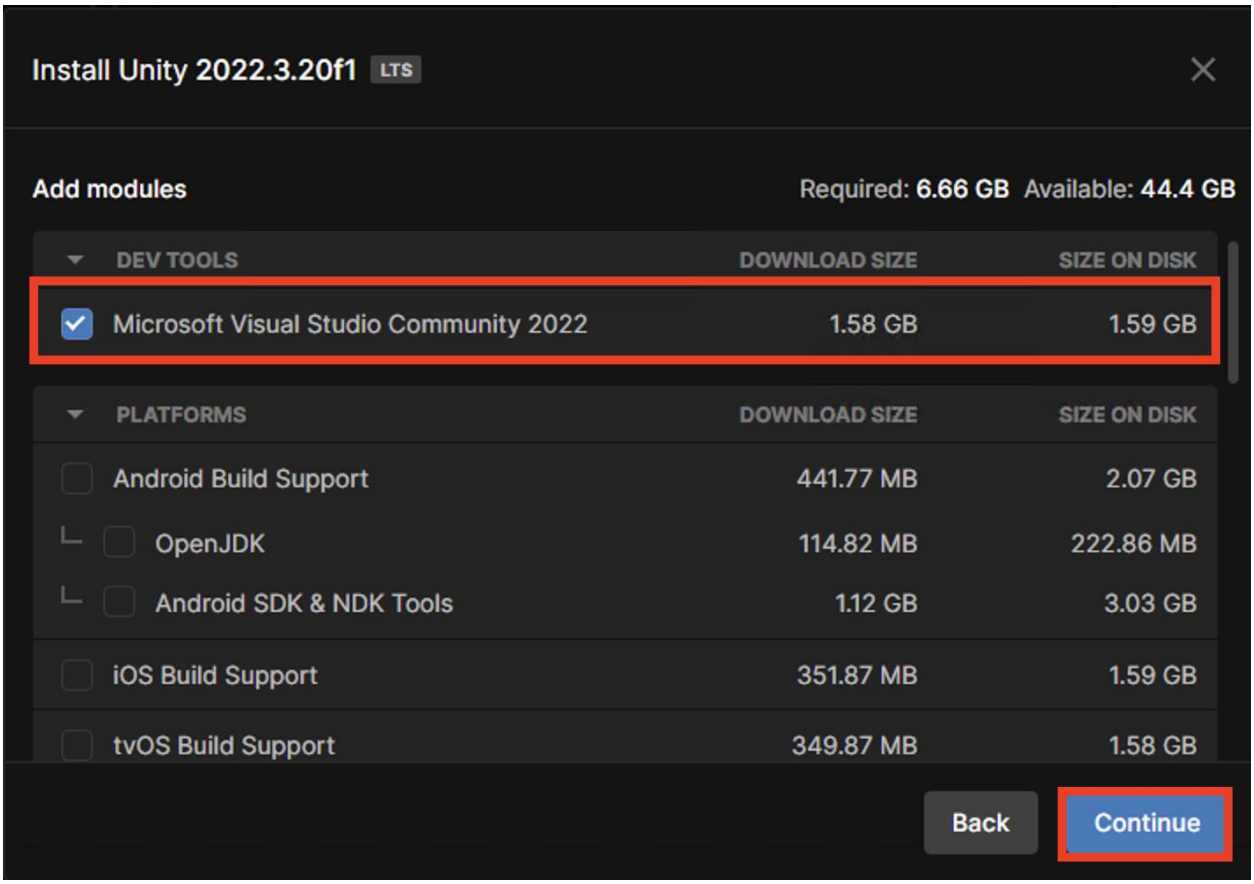


2. Once you select the installs tab, select the **Install Editor** button at the top. In the **Official Releases** window, select **Unity 2022.3 (LTS)**. Only the 2022.3 LTS version has been tested with Code Ninjas curriculum. The numbers after **2022.3** may not match the picture exactly but ensure that **LTS** is at the end of the picture.

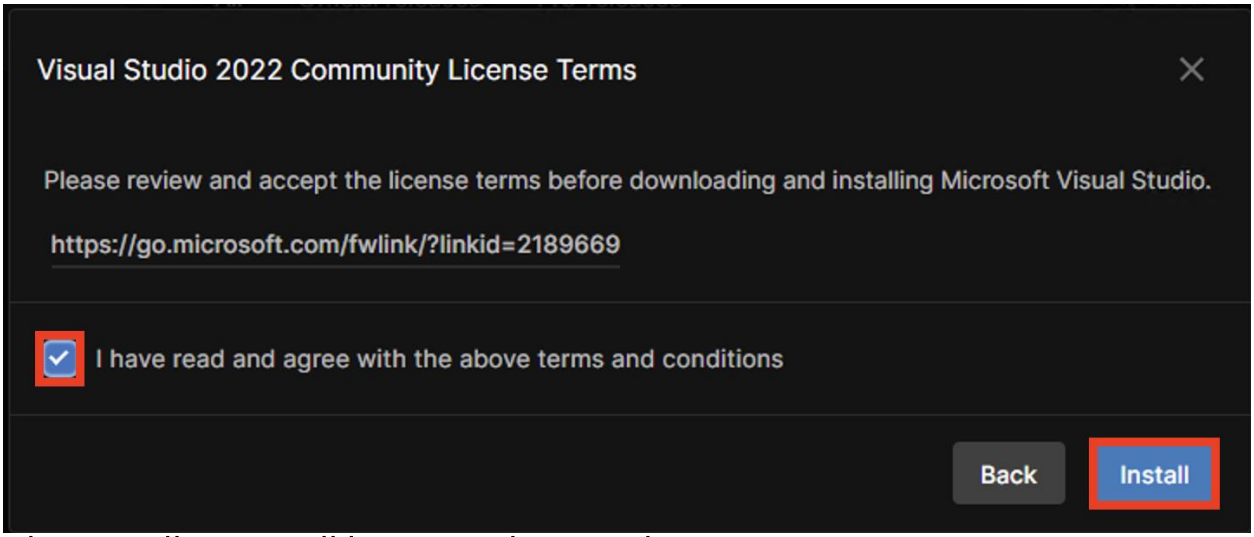
   Then, select **Install**.



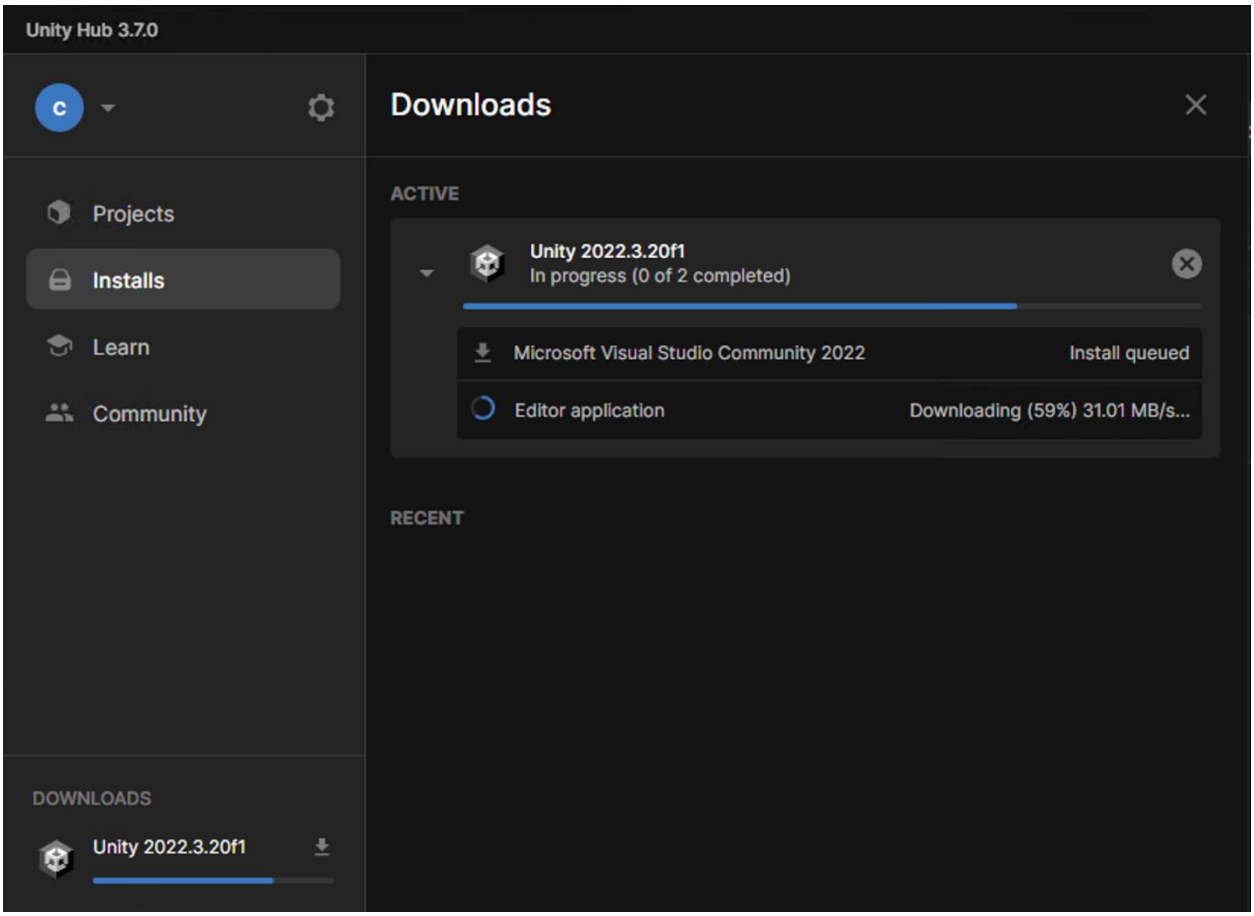
3. Select **Microsoft Visual Studio Community 2022** and click **Continue**.



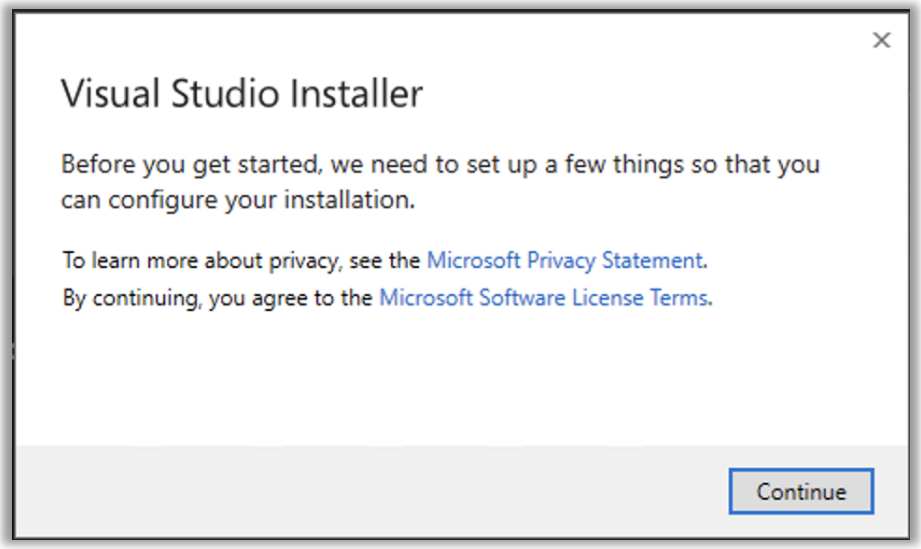
4. Accept the **Visual Studio** terms and conditions and click **Install**.



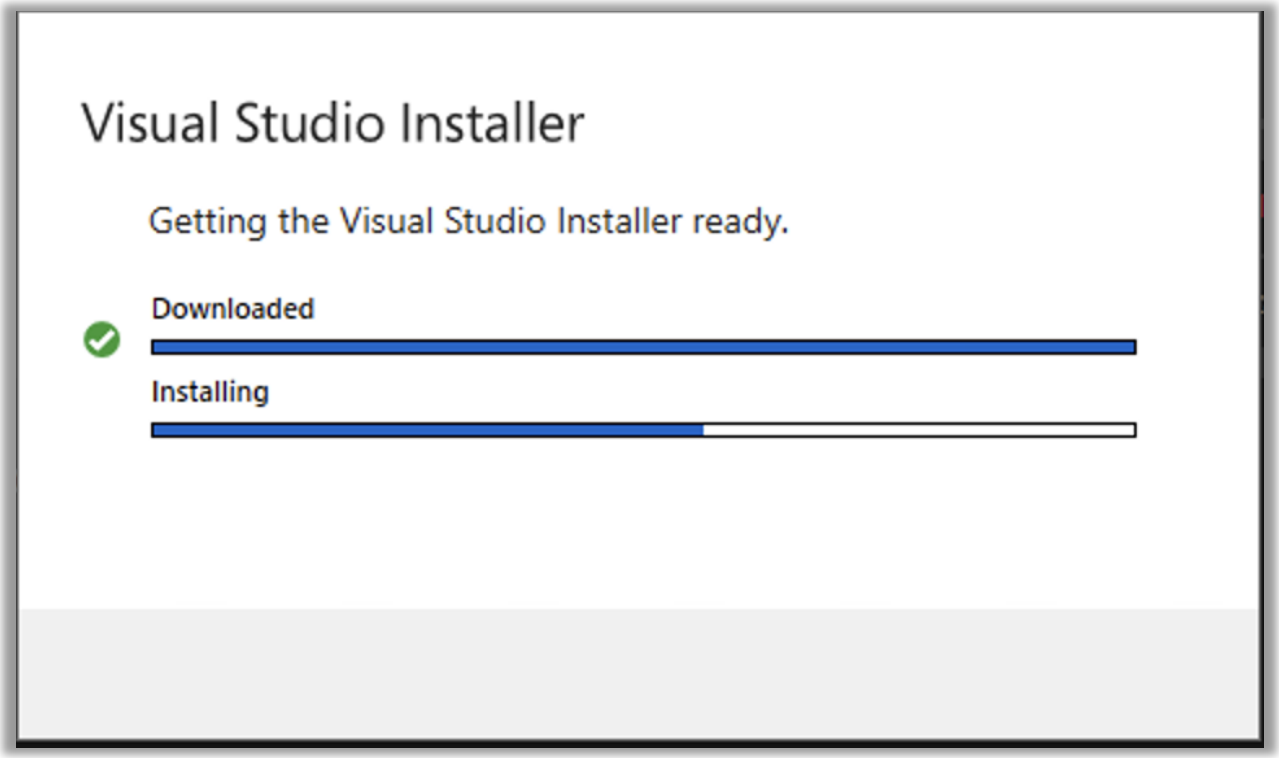
5. The installation will begin and may take 10 to 15 minutes depending on your Internet and computer speeds.



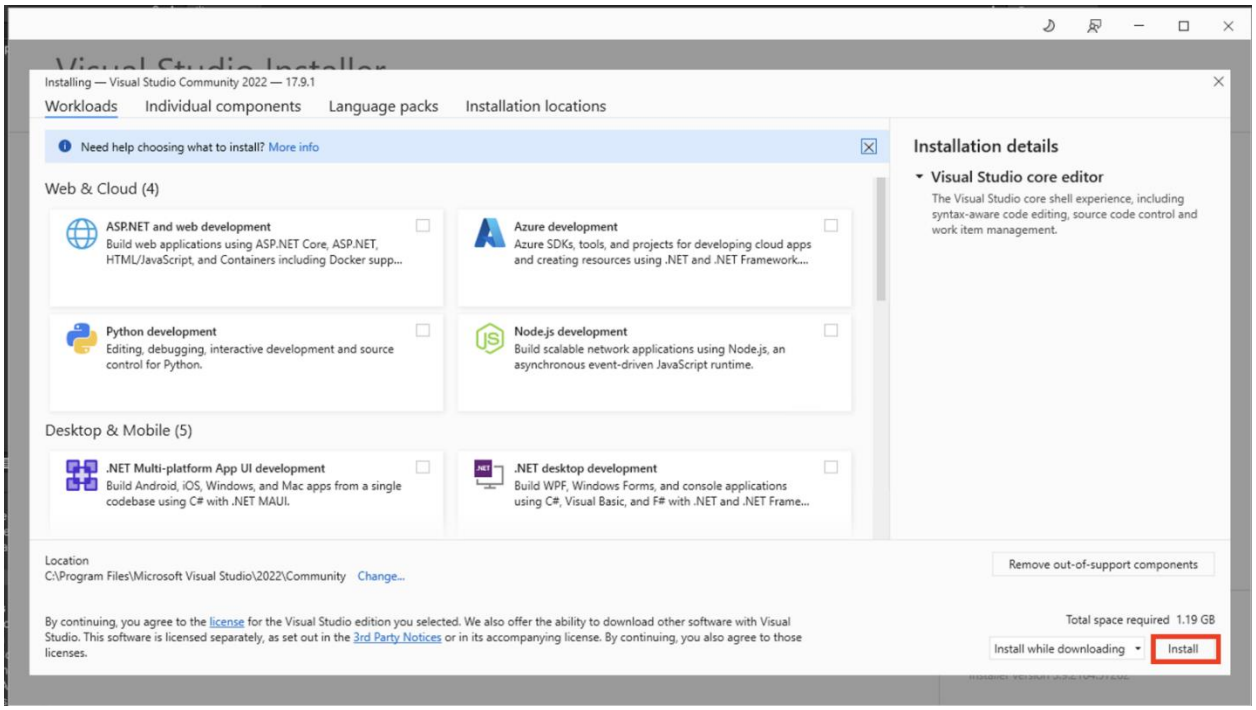
6. **Visual Studio** may require permission from you before it can begin to install. Select **Continue** if the popup below appears.



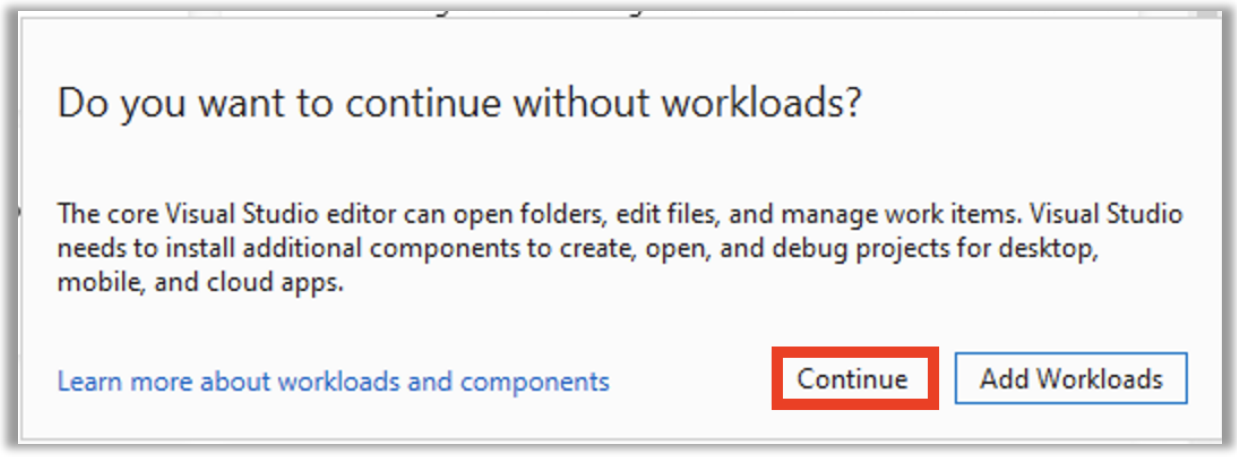
7. The **Visual Studio Installer** will begin if it is not already installed.



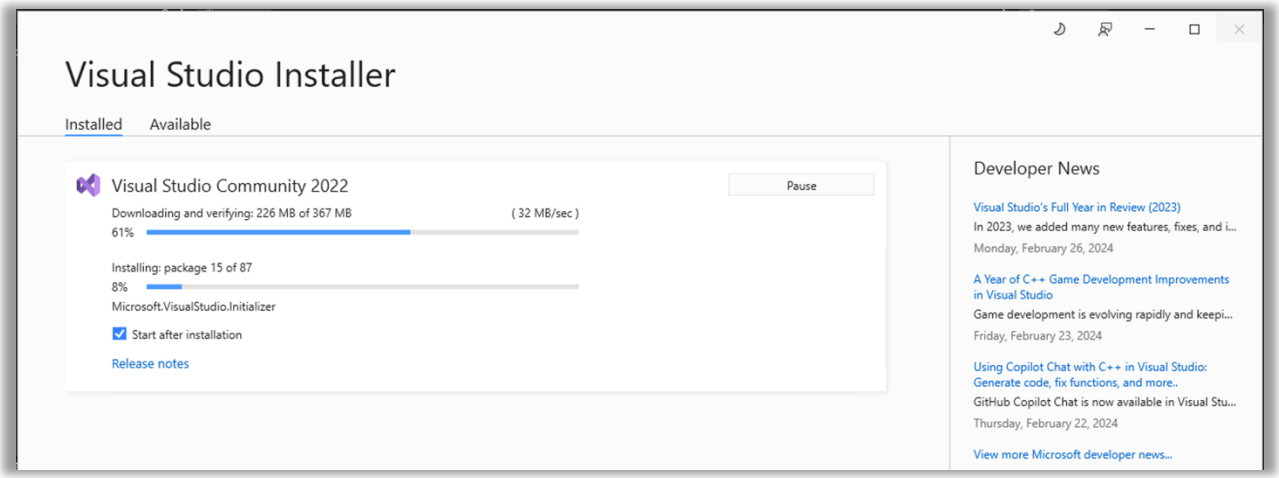
8. The **Visual Studio Installer** will appear, and it will preselect **Visual Studio Community 2022**. When the screen below appears, select the **Install** button.



9. A popup will appear when you select the **Install** button asking if you would like to continue without workloads. Select **Continue**.

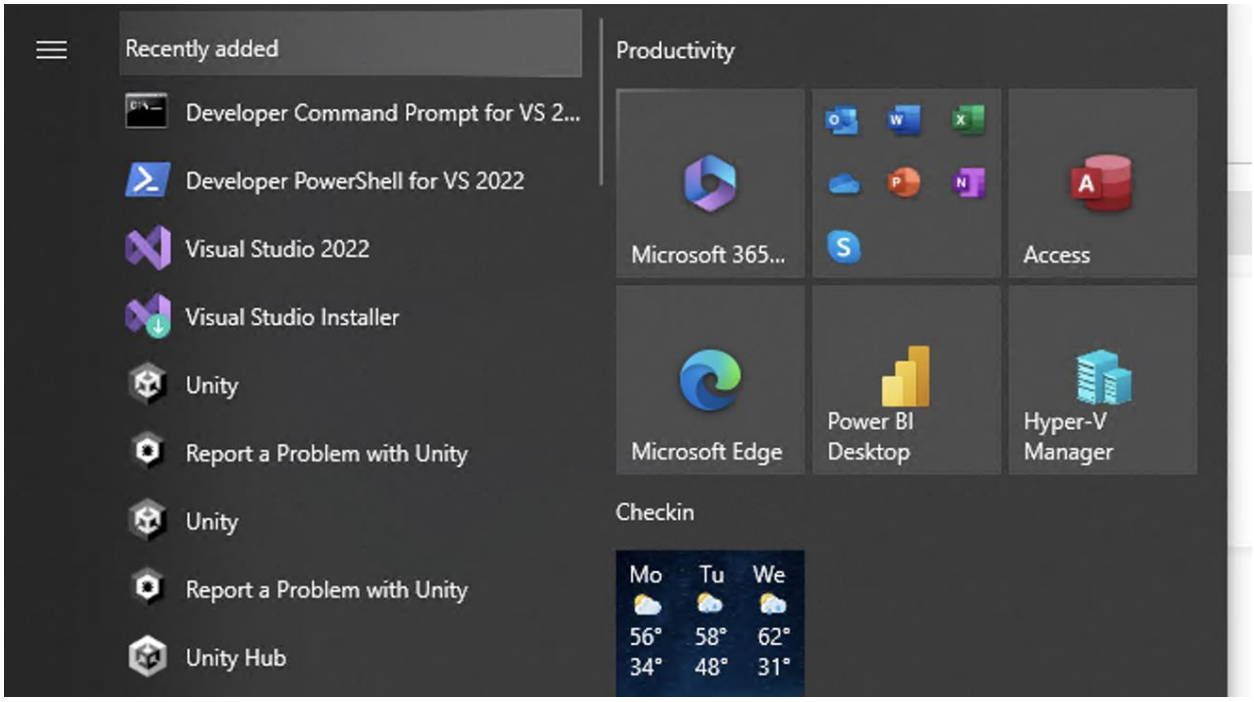


10. **Visual Studio Community 2022** will then begin to install.



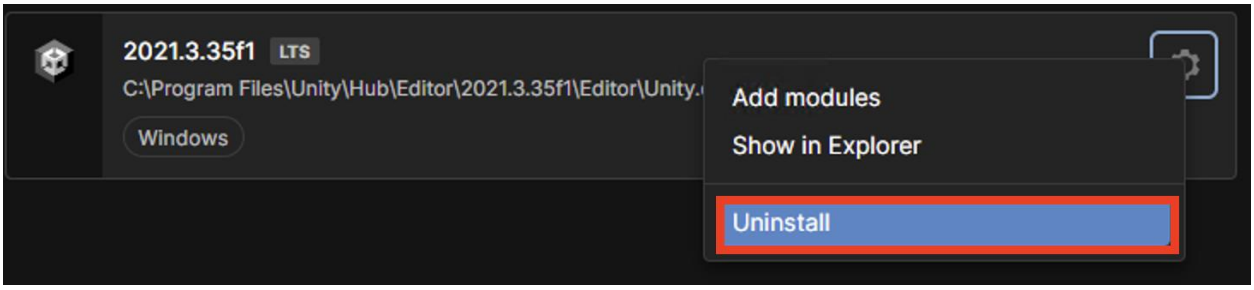
11. Once Visual Studio Community 2022 has installed it will ask you to sign in. Either create an account or sign in using a Microsoft 365 account.

12. After your computer restarts, verify that the installations were successful by finding the programs in the **Start** menu. If your computer is on Windows 11, this menu may look slightly different.



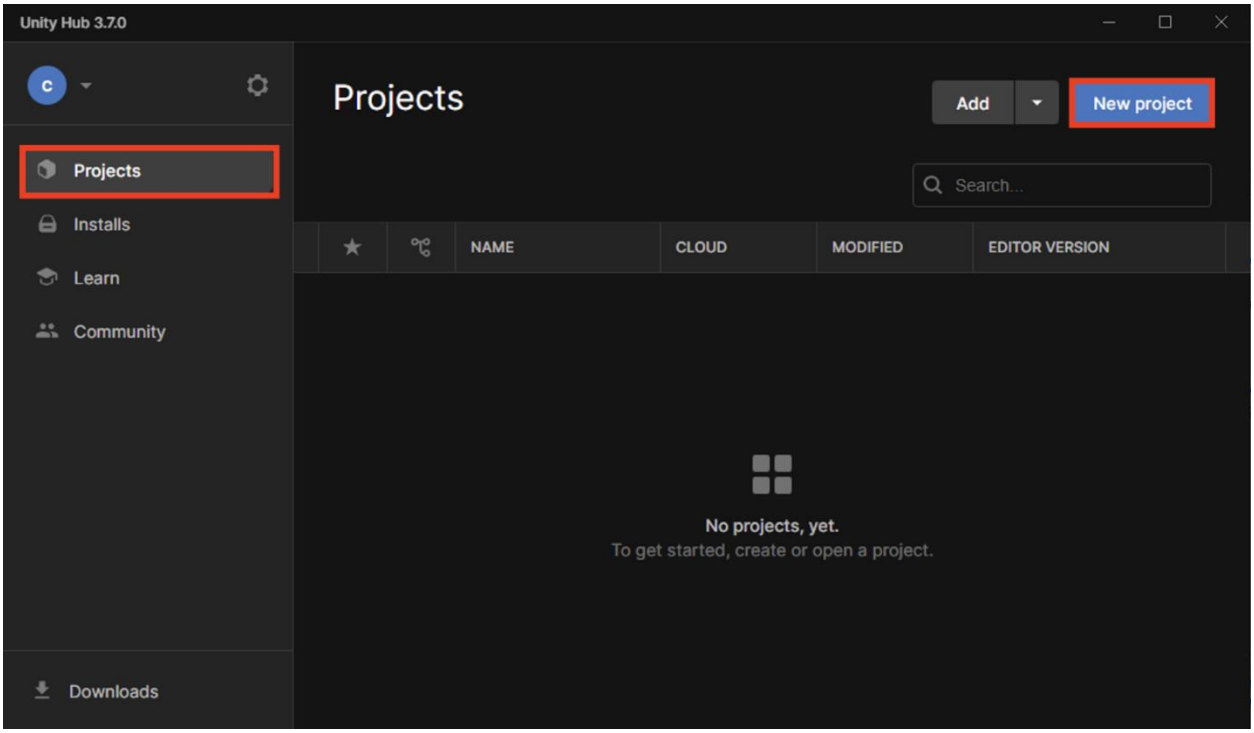
Removing Old Versions of Unity

Remove old versions of **Unity** by clicking the gear icon and selecting **Uninstall**. Depending on the way that the Unity version was installed on your computer, you might need to locate the installation's folder on your computer and delete it manually.

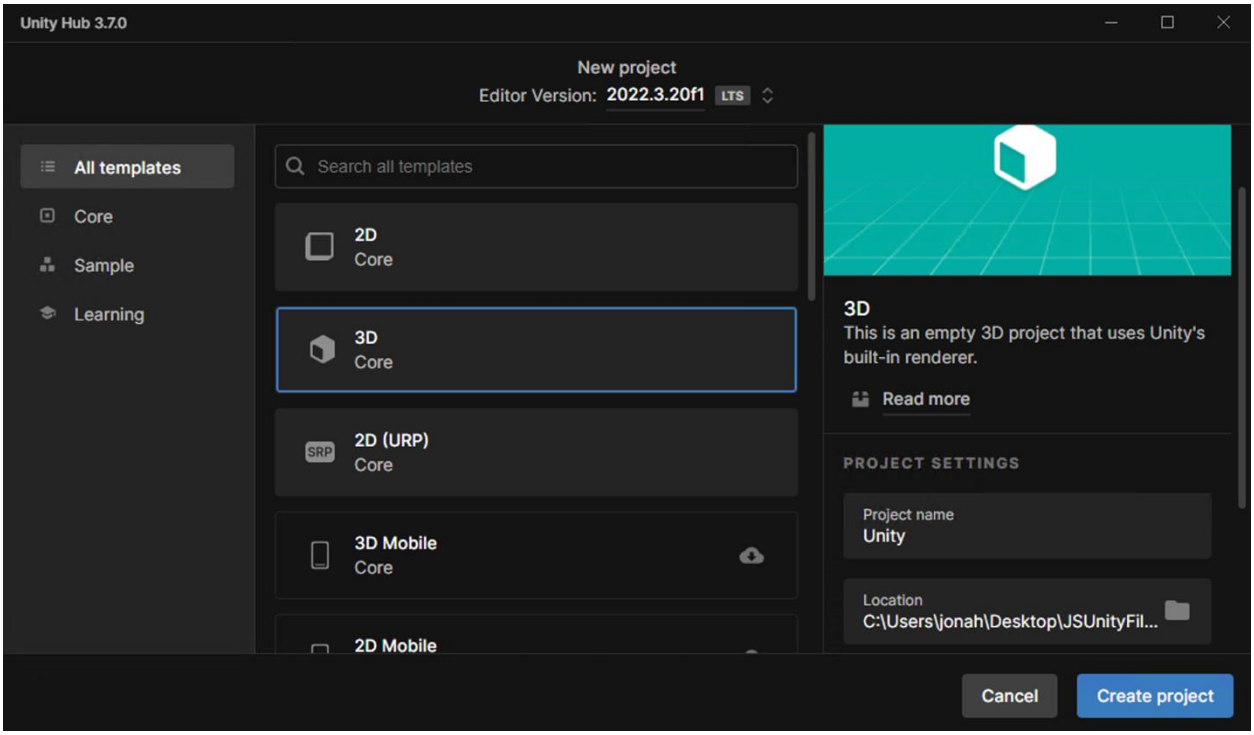


Creating a New Unity Project

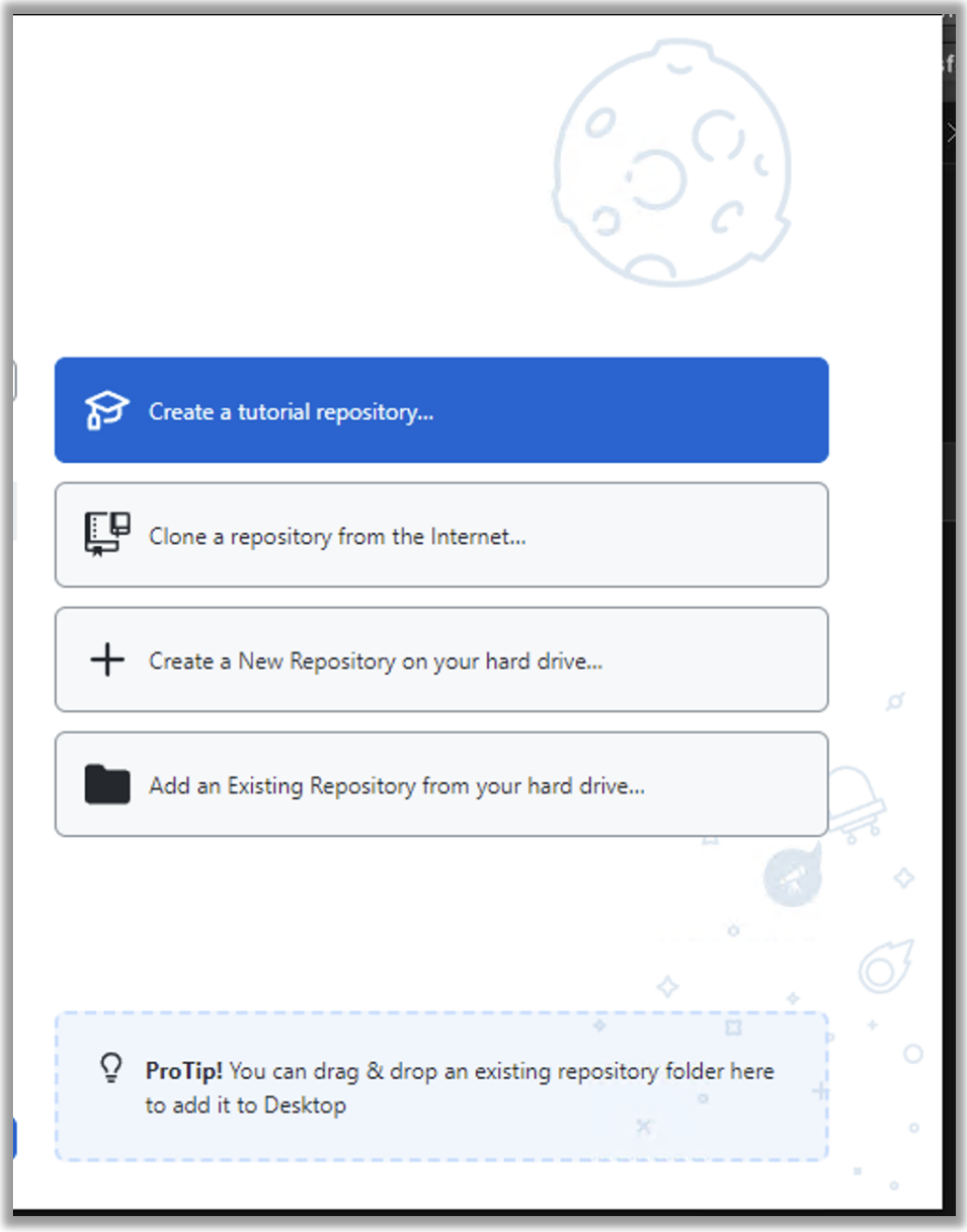
Open the **Projects** tab and click the **New Project** button to start a new **Unity project**.

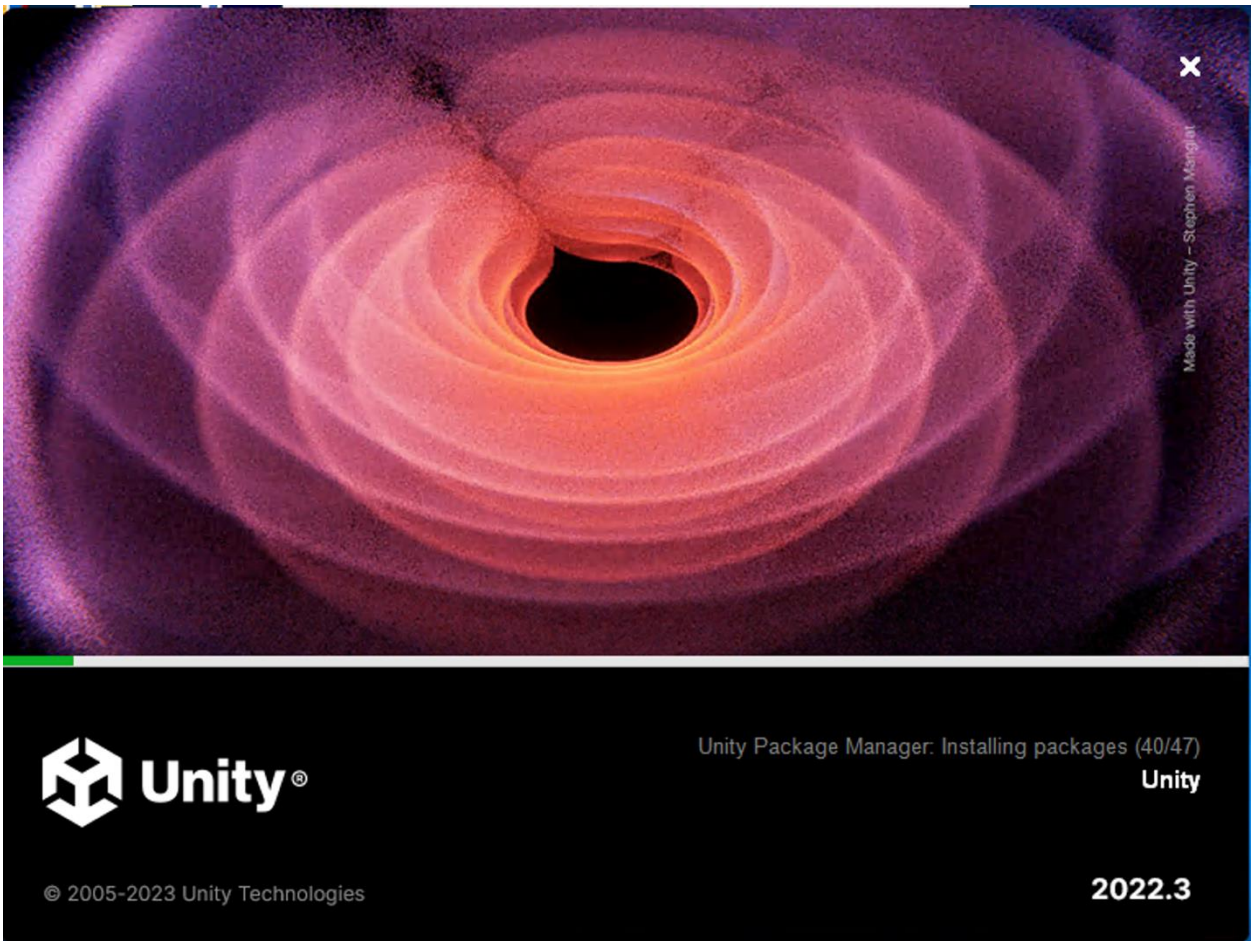


1. Give the project a name and specify a location that is easily accessible. Ensure that the Editor Version at the top reads 2022.3. The templates window in the middle is where you will select between 2D and 3D.

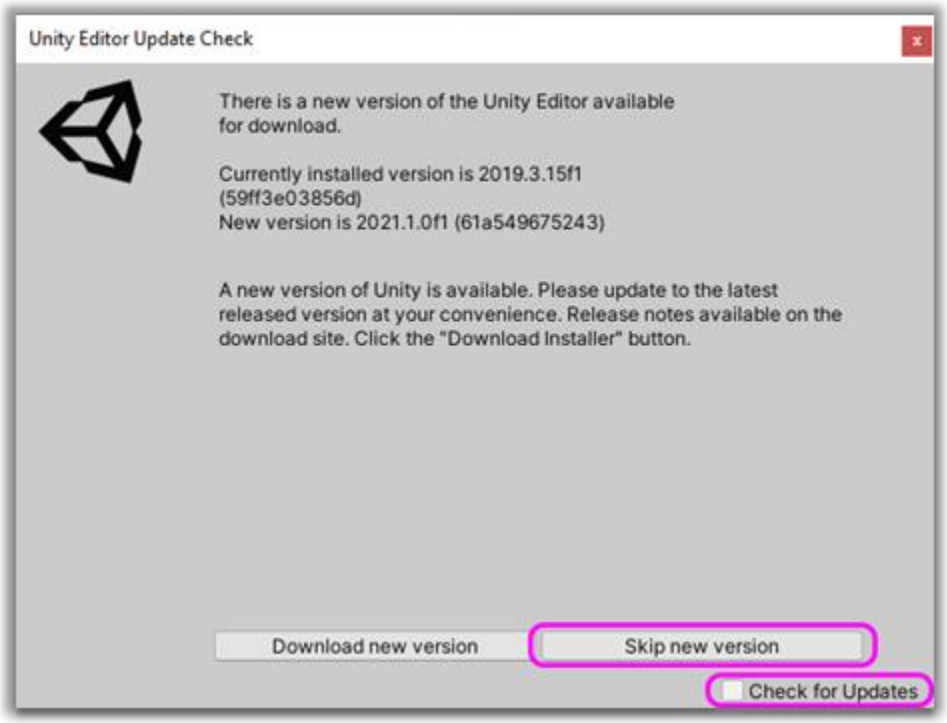


2. Wait for **Unity** to initialize your project.

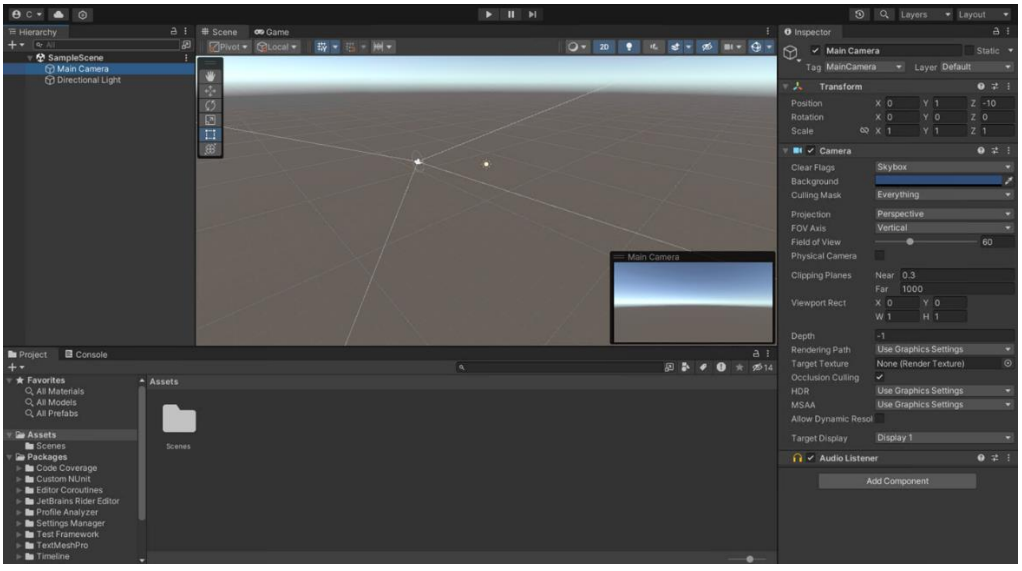




3. To ensure compatibility with the curriculum, uncheck **Check for Updates** and click **Skip new version**.

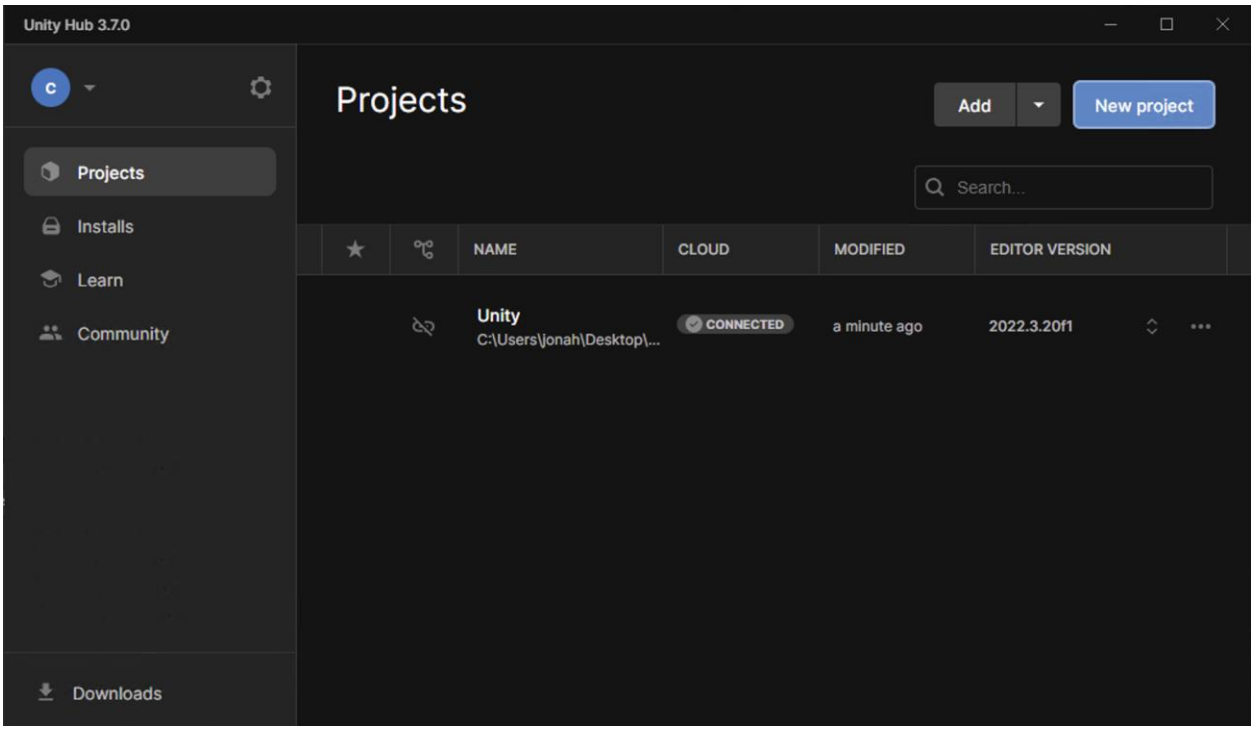


4. The **Unity editor** is where Ninjas will program their games.

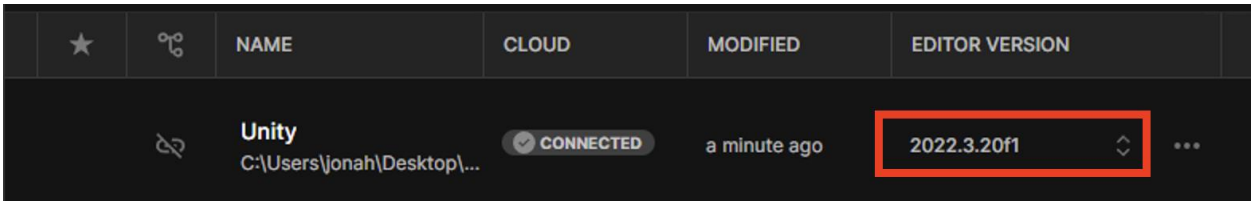


Opening an Existing Project

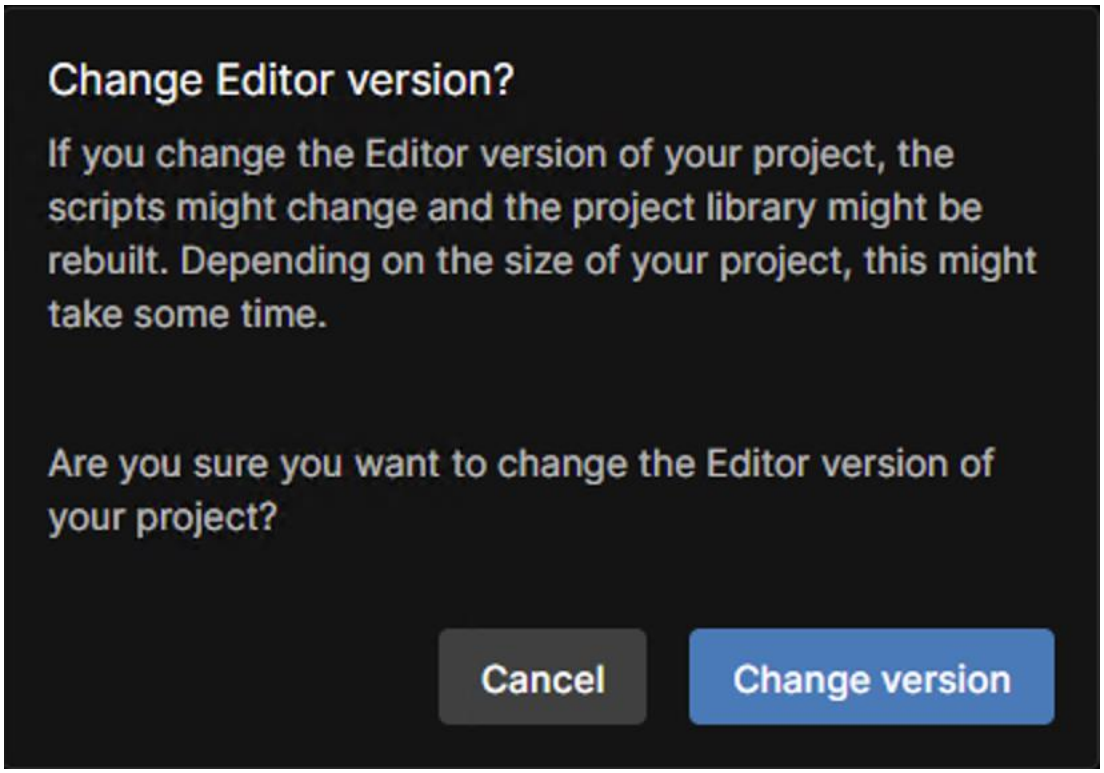
You can view a list of projects in **Unity Hub**.



Each **Project** is locked to a specific version, but you can use the drop down to change it.



When switching to a newer version, this popup will appear. To change versions, select **Change Version**.



To open the project, click the project's entry in the list.

GitHub

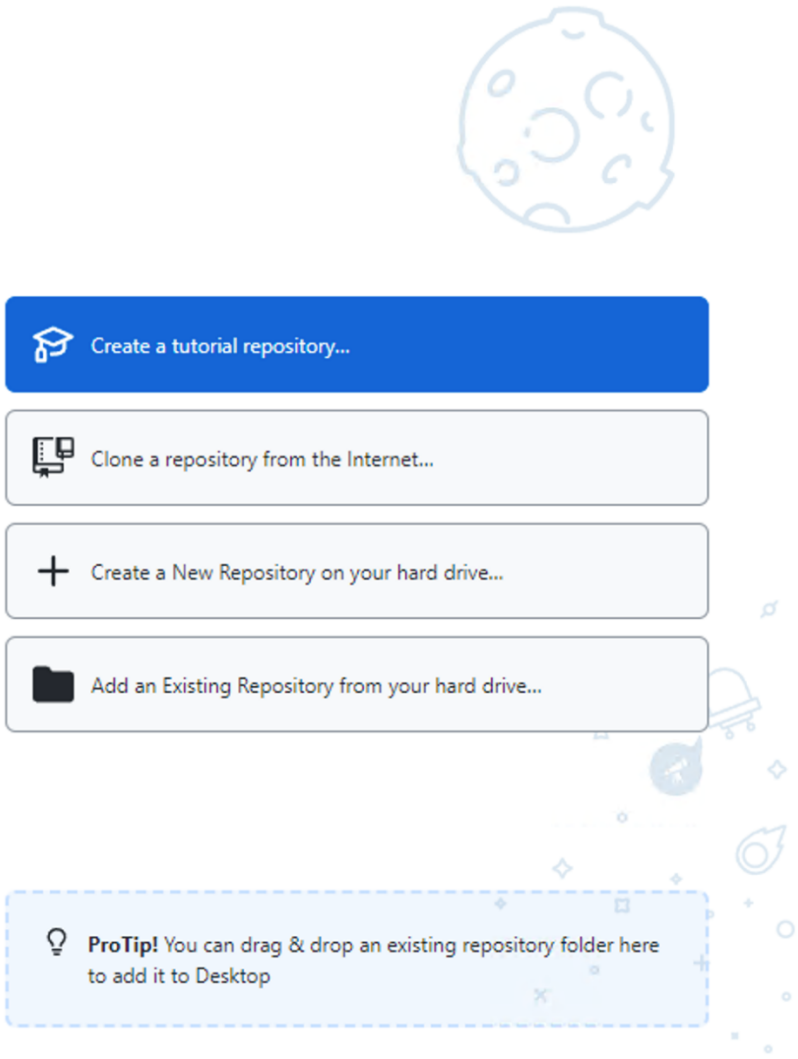
GitHub can be used to save projects to the cloud so they can be synced to more than one computer. GitHub works on a per-account basis, so you can either use one account per center or per ninja.

Download and install **GitHub Desktop** from https://desktop.github.com/.

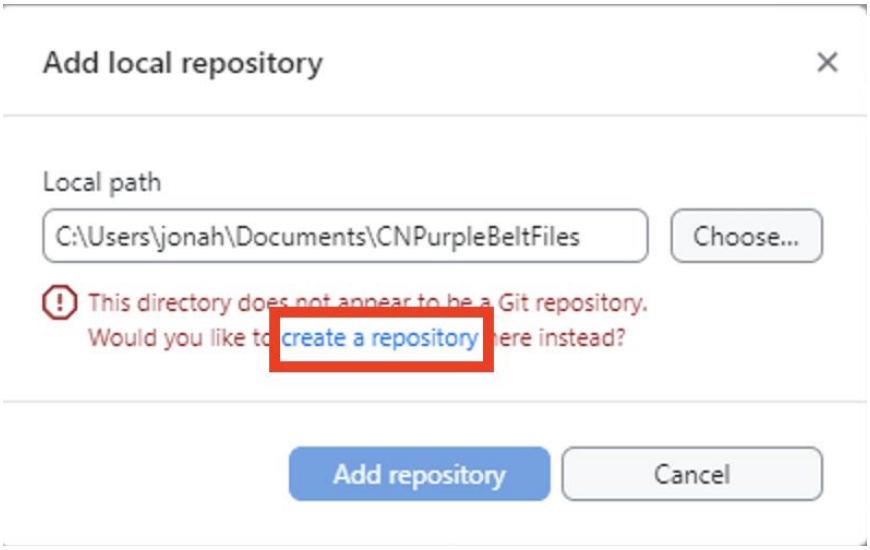
Run the program and log in. Your screen will show this menu.

Creating a New GitHub Repository

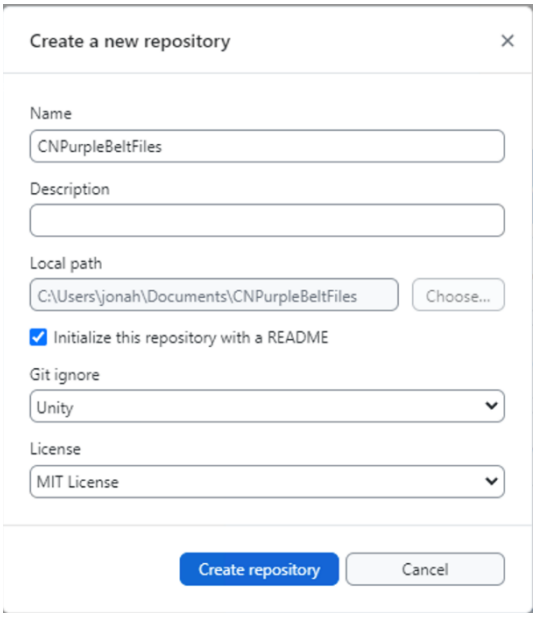
Select **Add an Existing Repository from your hard drive.** Then select the folder where your Unity files are stored.



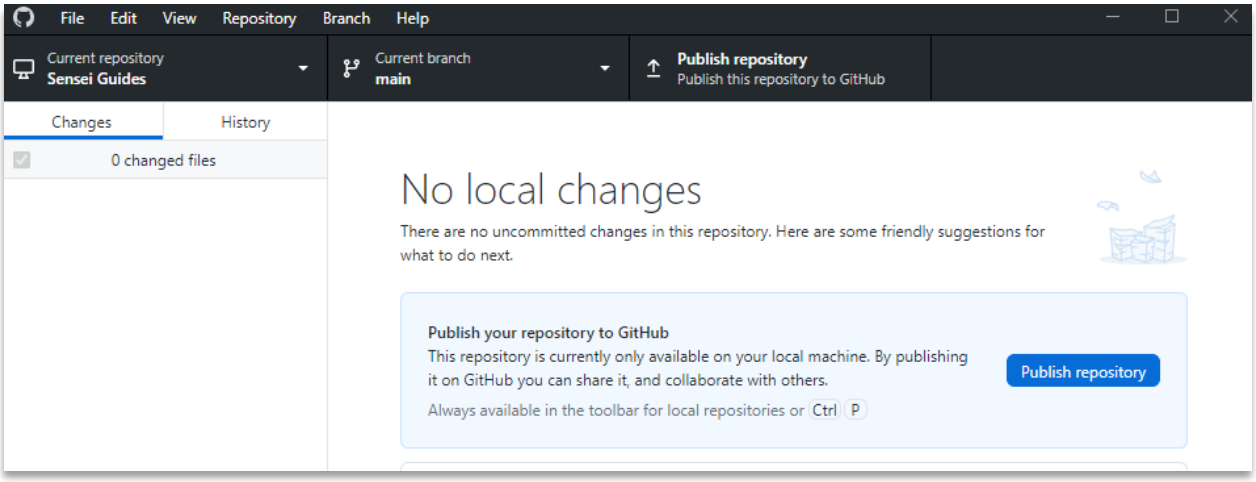
Click the **create a repository** button.



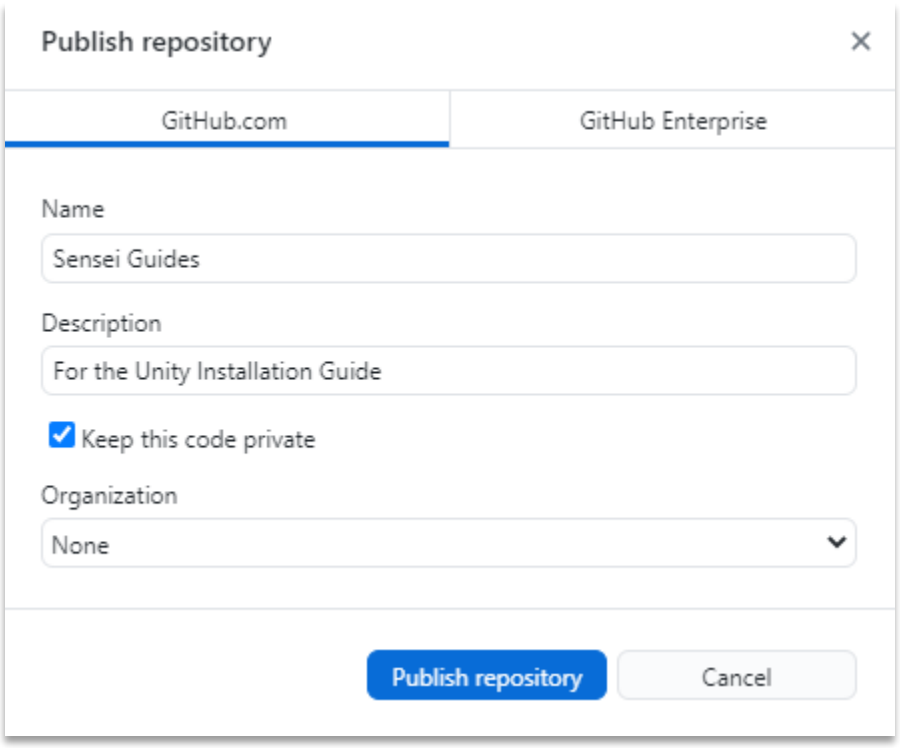
Fill out the **description** and select **Initialize this repository with a README**. Be sure to set the **Git ignore** to **Unity** and select a **license**. Click **Create repository**.



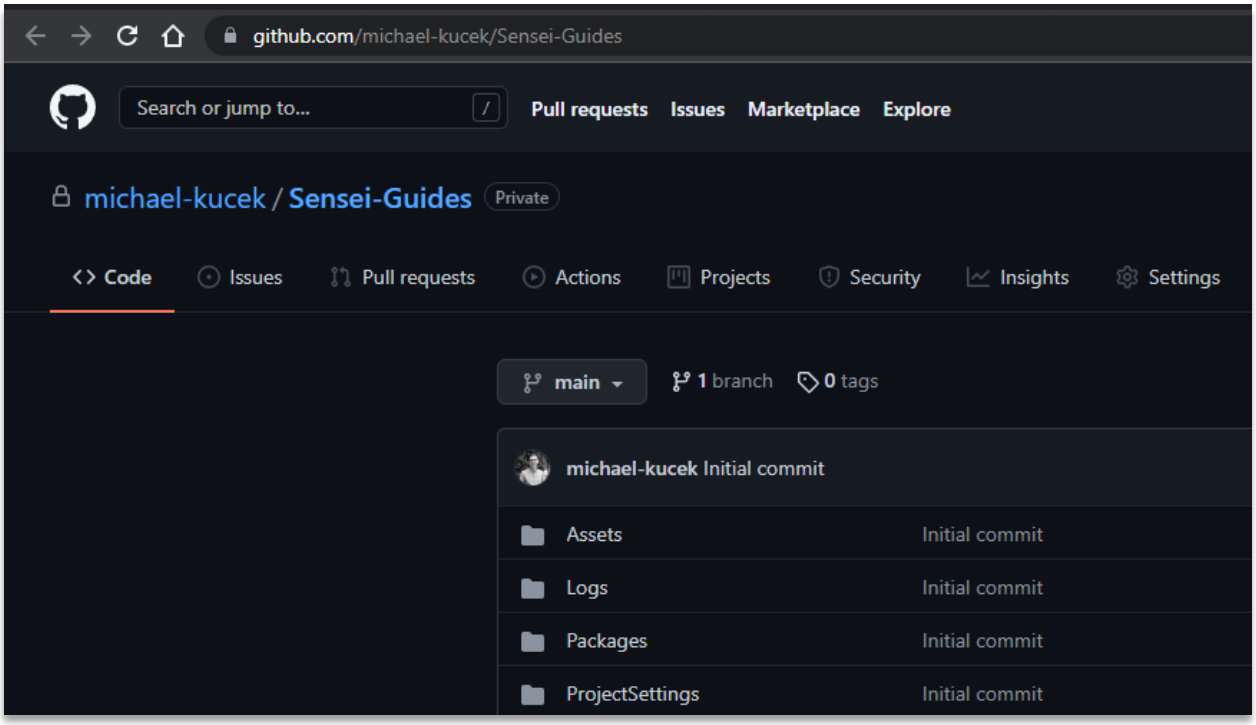
All of the files located inside the selected project folder will be scanned and added to the local git repository. Click **Publish repository** to publish the project to the GitHub account.



You can choose to keep the code private or make it public. Click **Publish repository**.

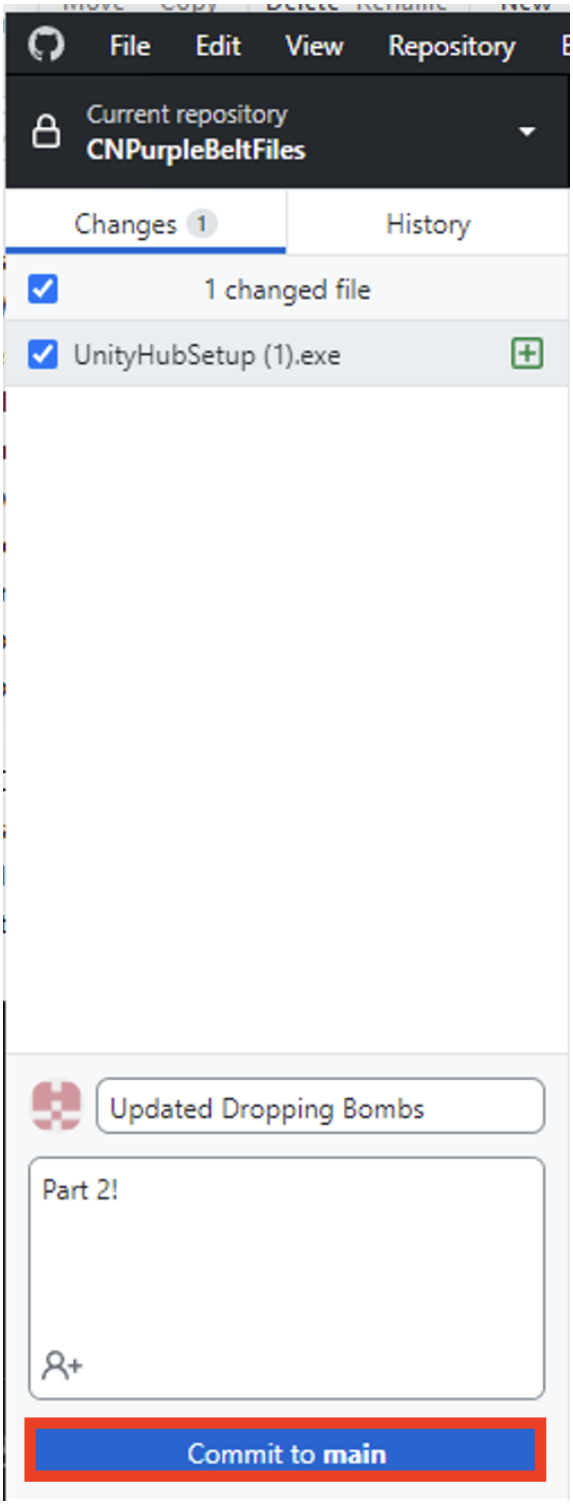


The project is now published and can be downloaded on any other computer logged in to the correct GitHub account.

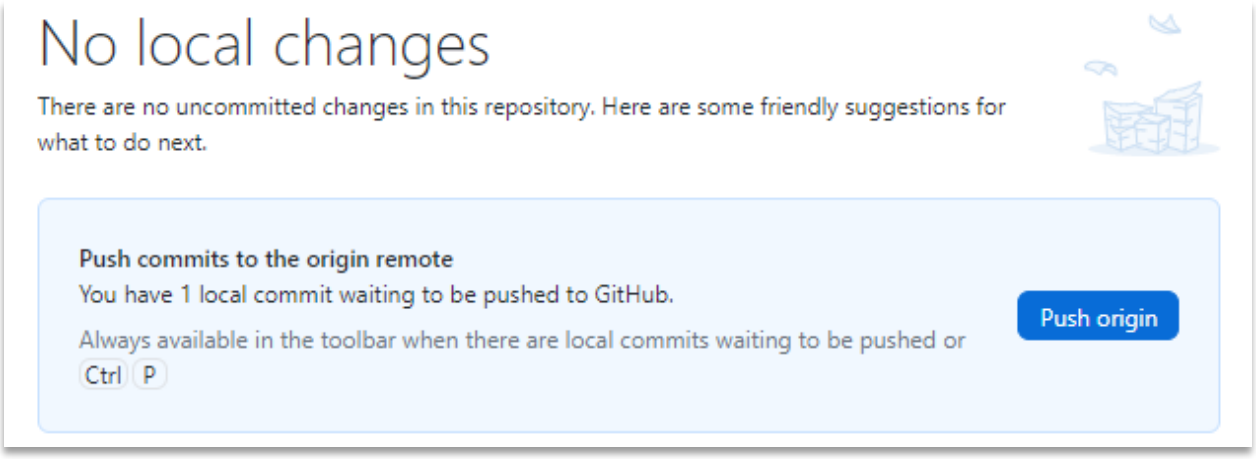


Syncing Local Changes to GitHub

Once changes are made to the project, you need to manually sync them with GitHub Desktop. With the project open, you will see a list of all the unsaved changes. Give the changes a **summary** and an optional **description**. Click **Commit to main**.



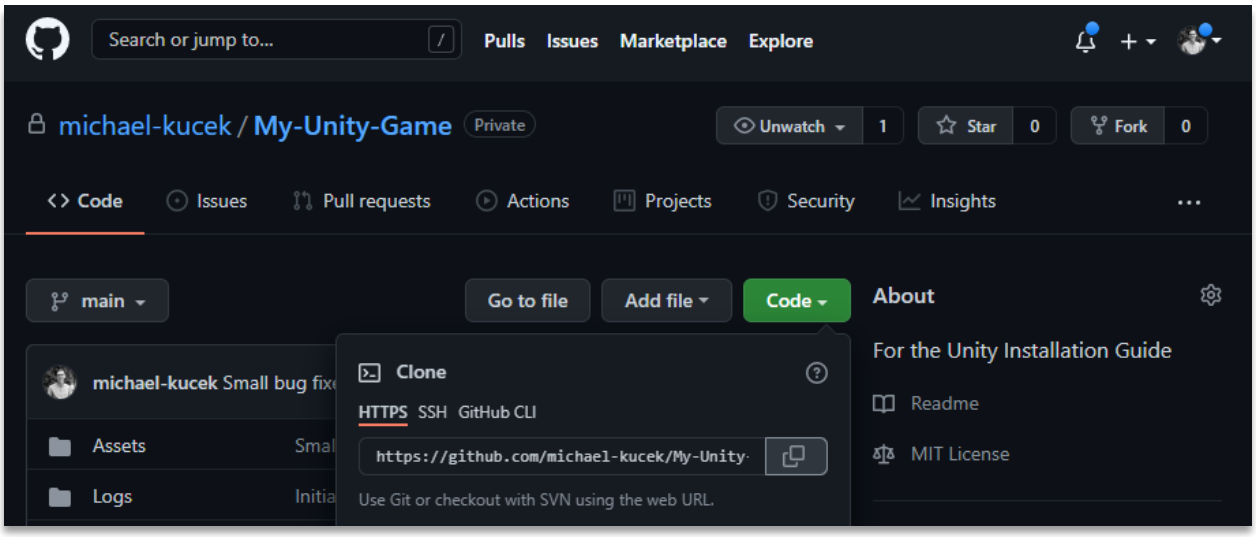
These changes are now saved to your local repository. To save them to GitHub, you need to click **Push origin**.



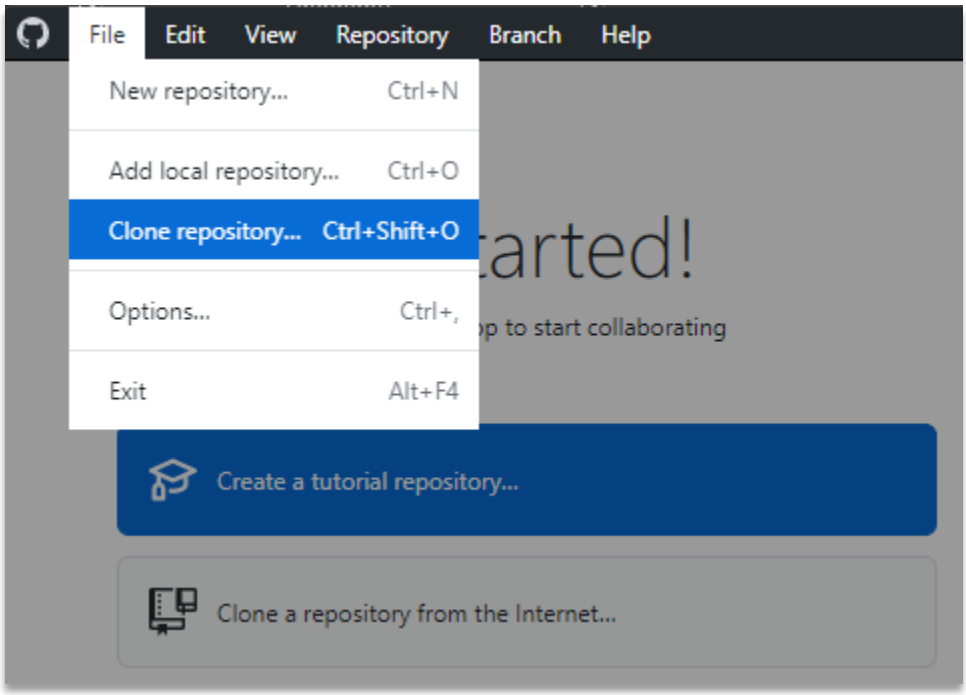
This process needs to be repeated every time changes are made.

Syncing an Existing GitHub Repository

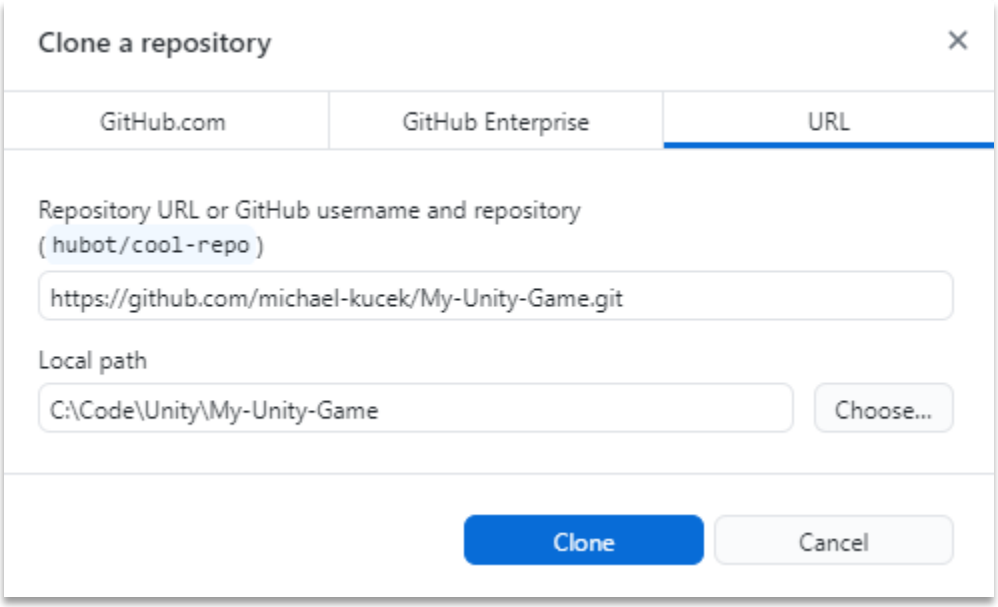
To download an existing GitHub project to a new computer, you need the project's **GitHub URL**. Navigate to the GitHub project page, click on **Code**, and then copy the URL.



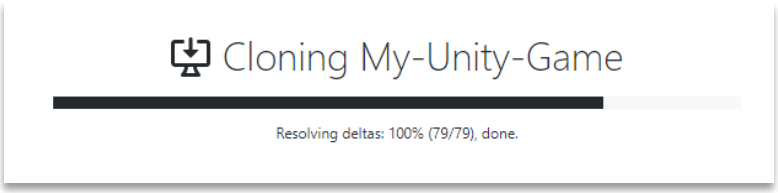
In GitHub Desktop, select **Clone a repository**.

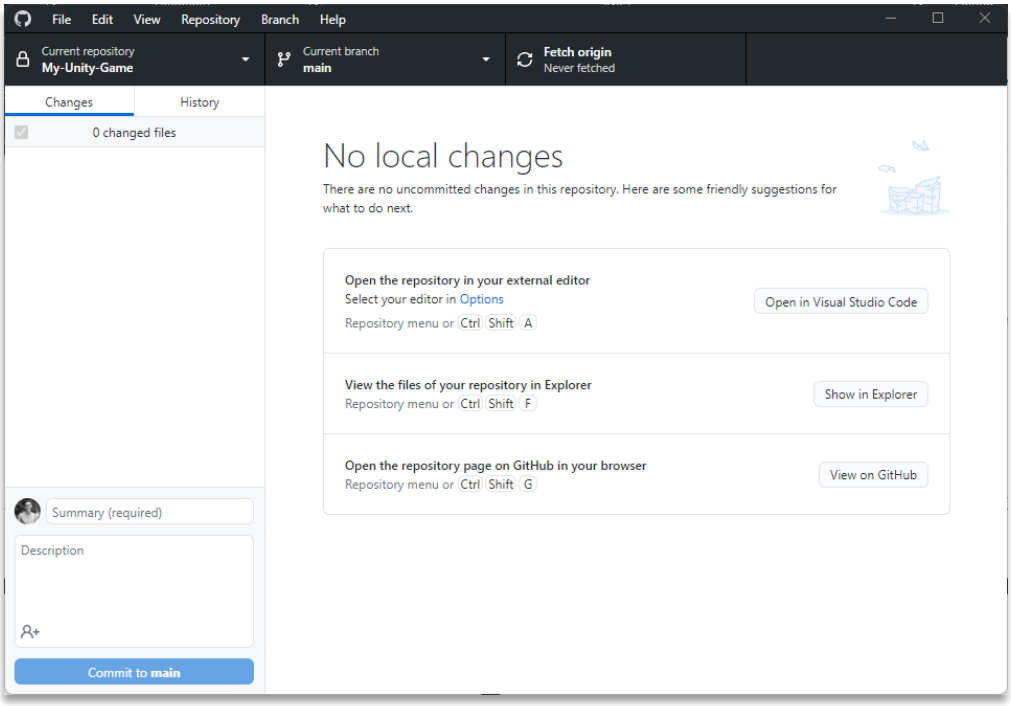


Paste in the **GitHub URL** and choose where you want to save the local repository. Click **Clone**.



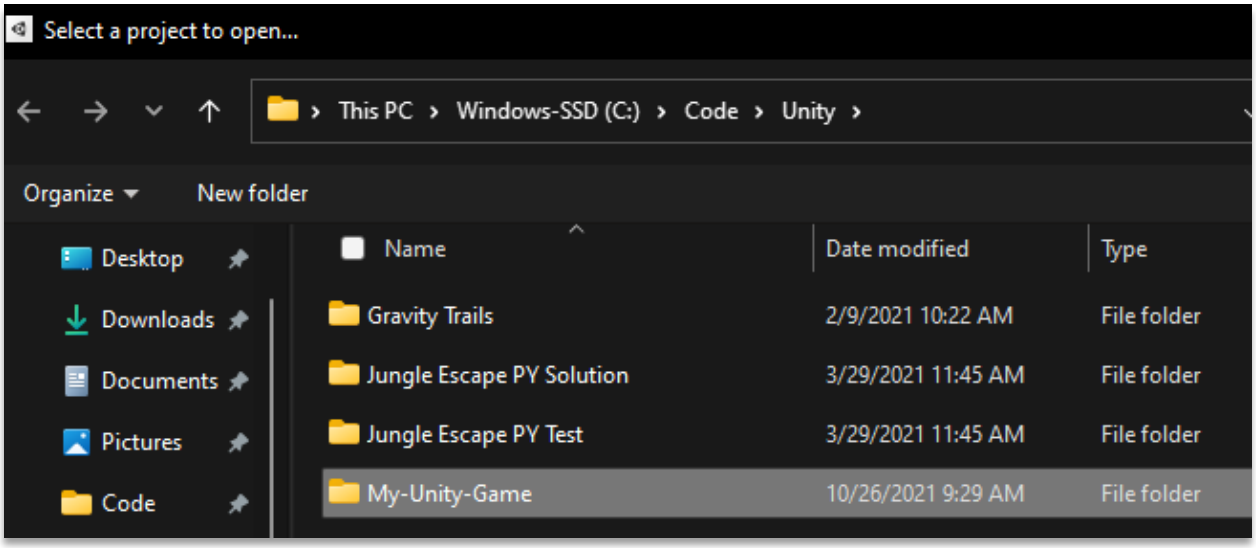
The project will be synced locally to the computer.

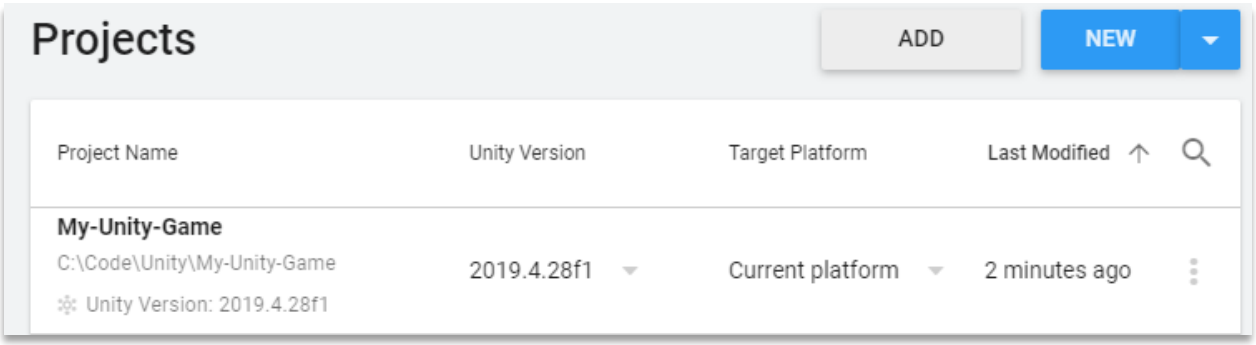




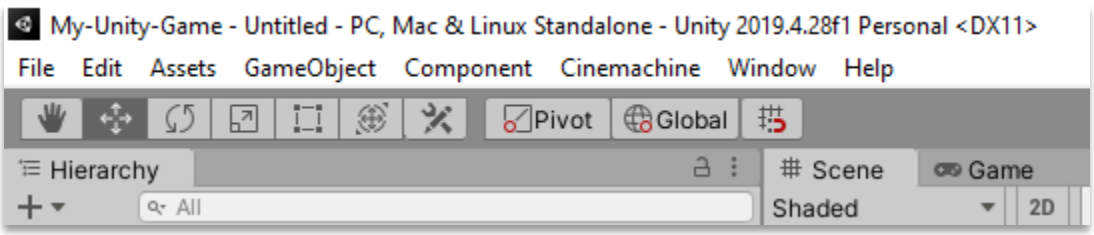
Open **Unity Hub** and click **Add**.

Find the project that was just downloaded from GitHub.



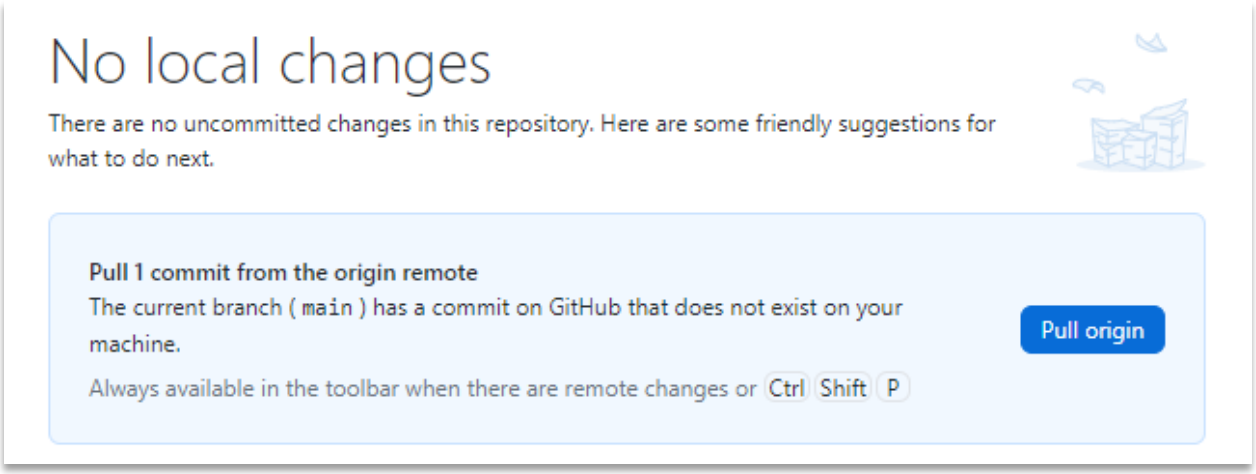


Open the project in Unity. Be sure to sync any changes.



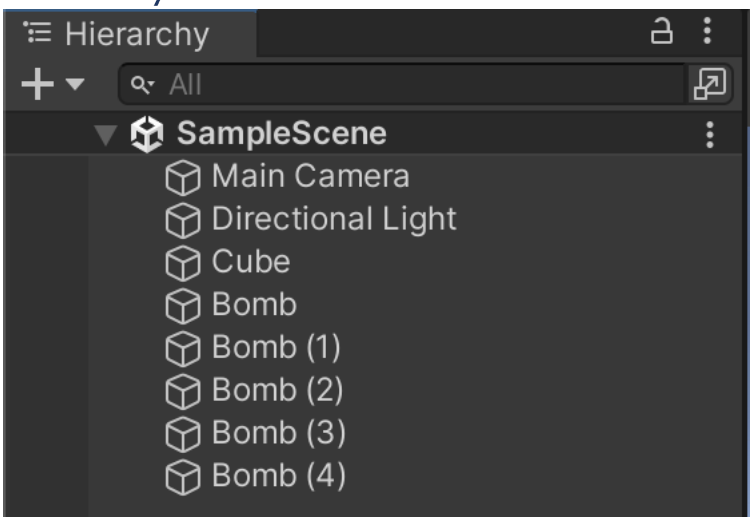
Syncing Changes from Another Computer

If changes to the game have been made on another computer, you must manually sync the changes using GitHub Desktop. To ensure your local copy is synced with the GitHub project, click Pull origin when GitHub detects your local version is out of date.

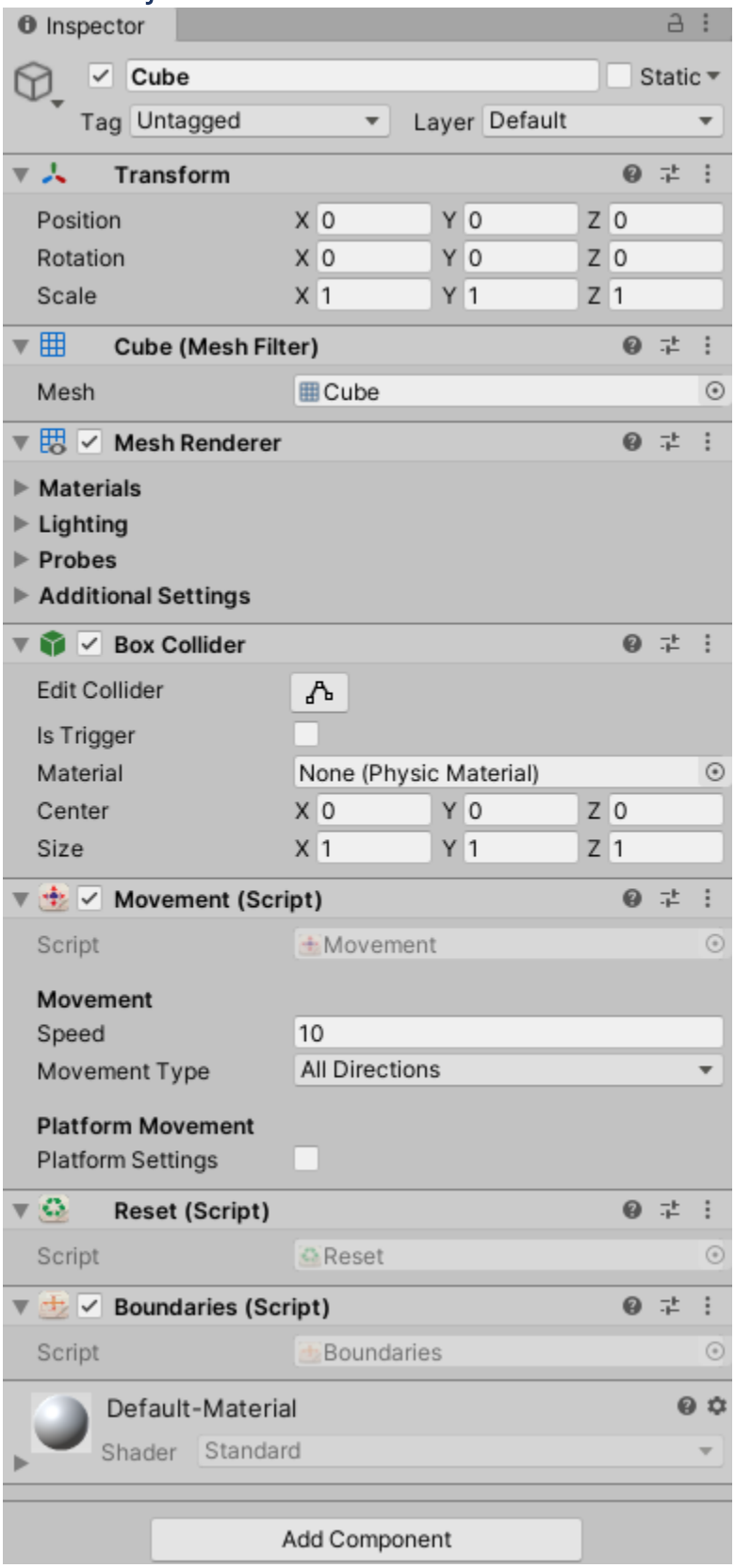


Activity Solution: Dropping Bombs

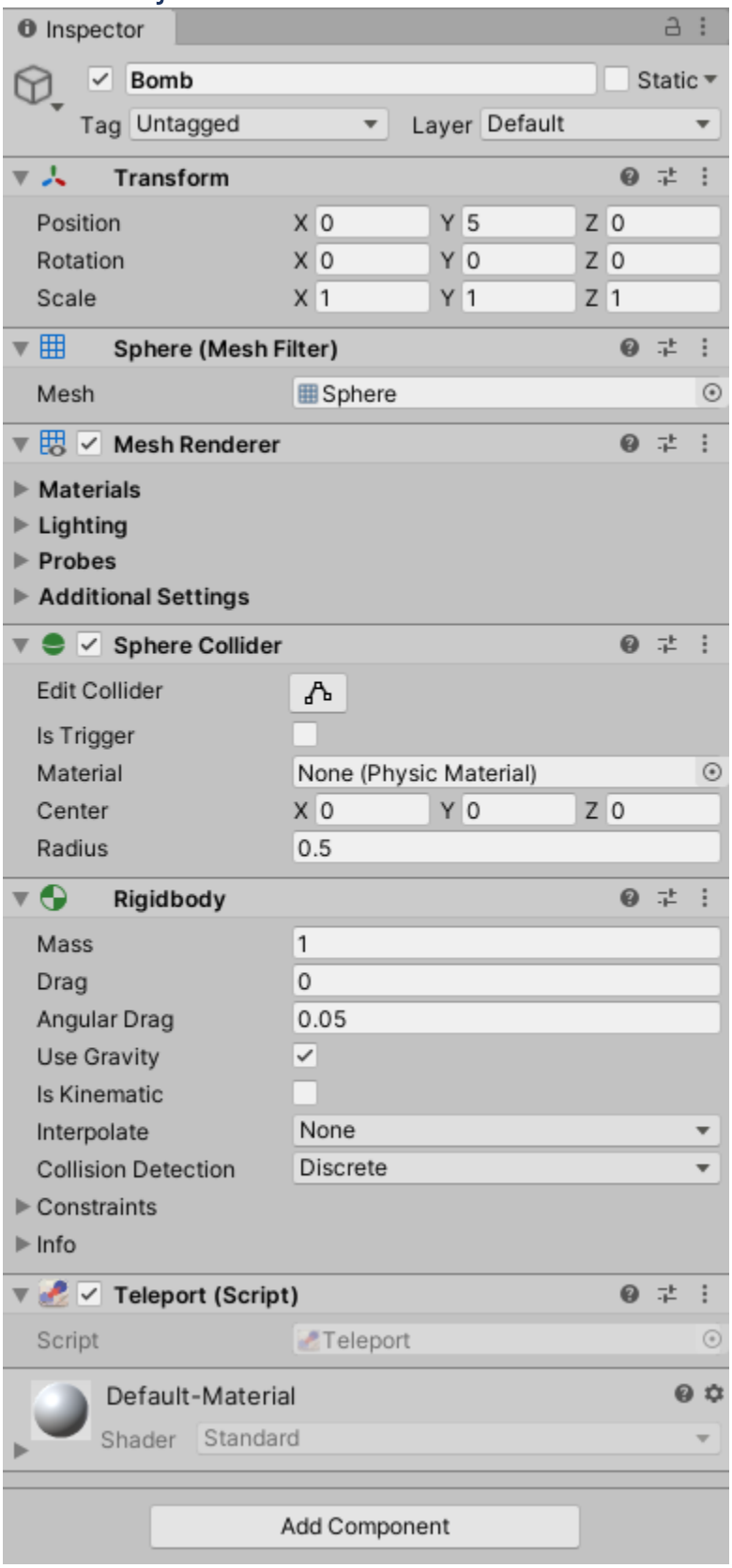
Hierarchy



Cube Object

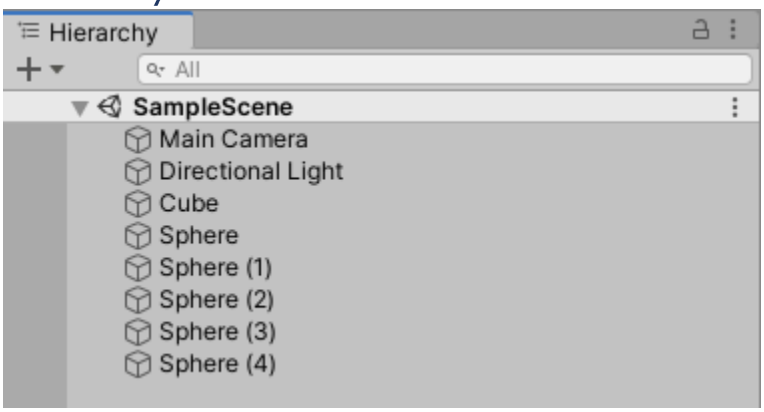


Bomb Object



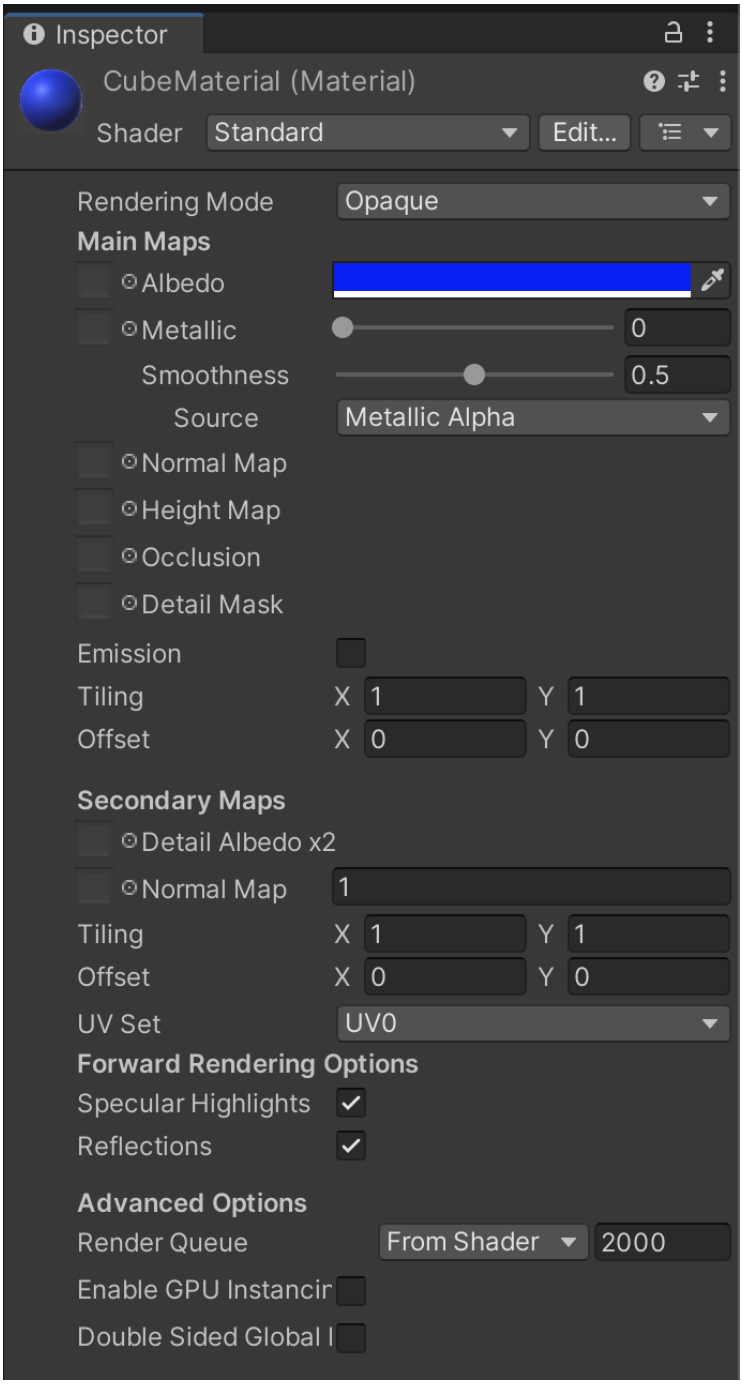
Activity Solution: Color Drop Prove Yourself

Hierarchy



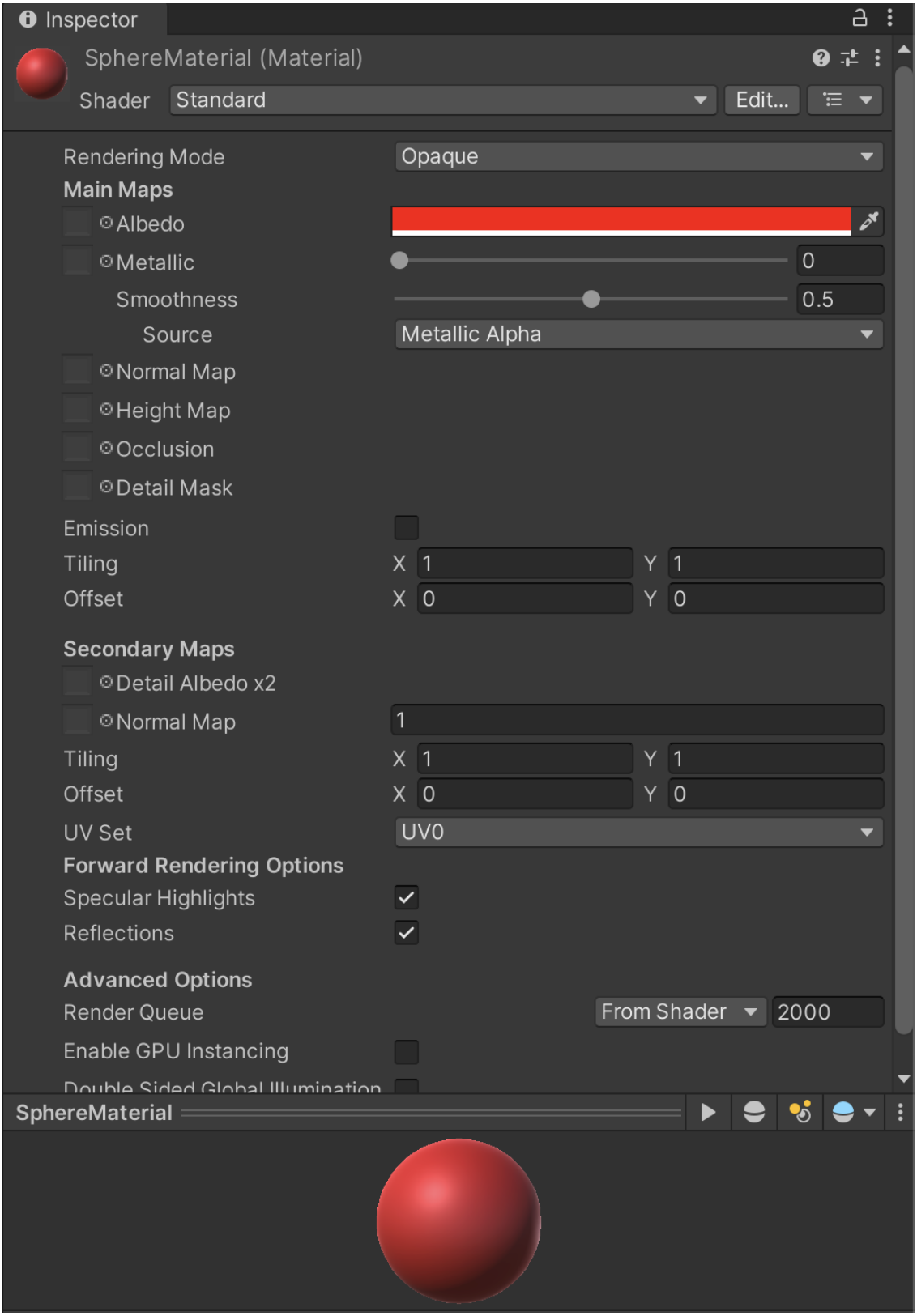
Cube Material

Ninjas can select use any color.

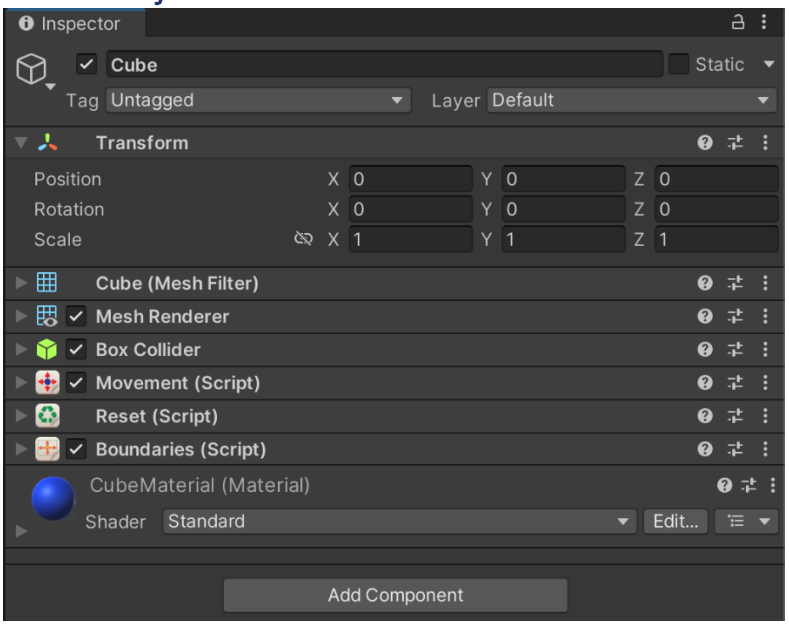


Sphere Material

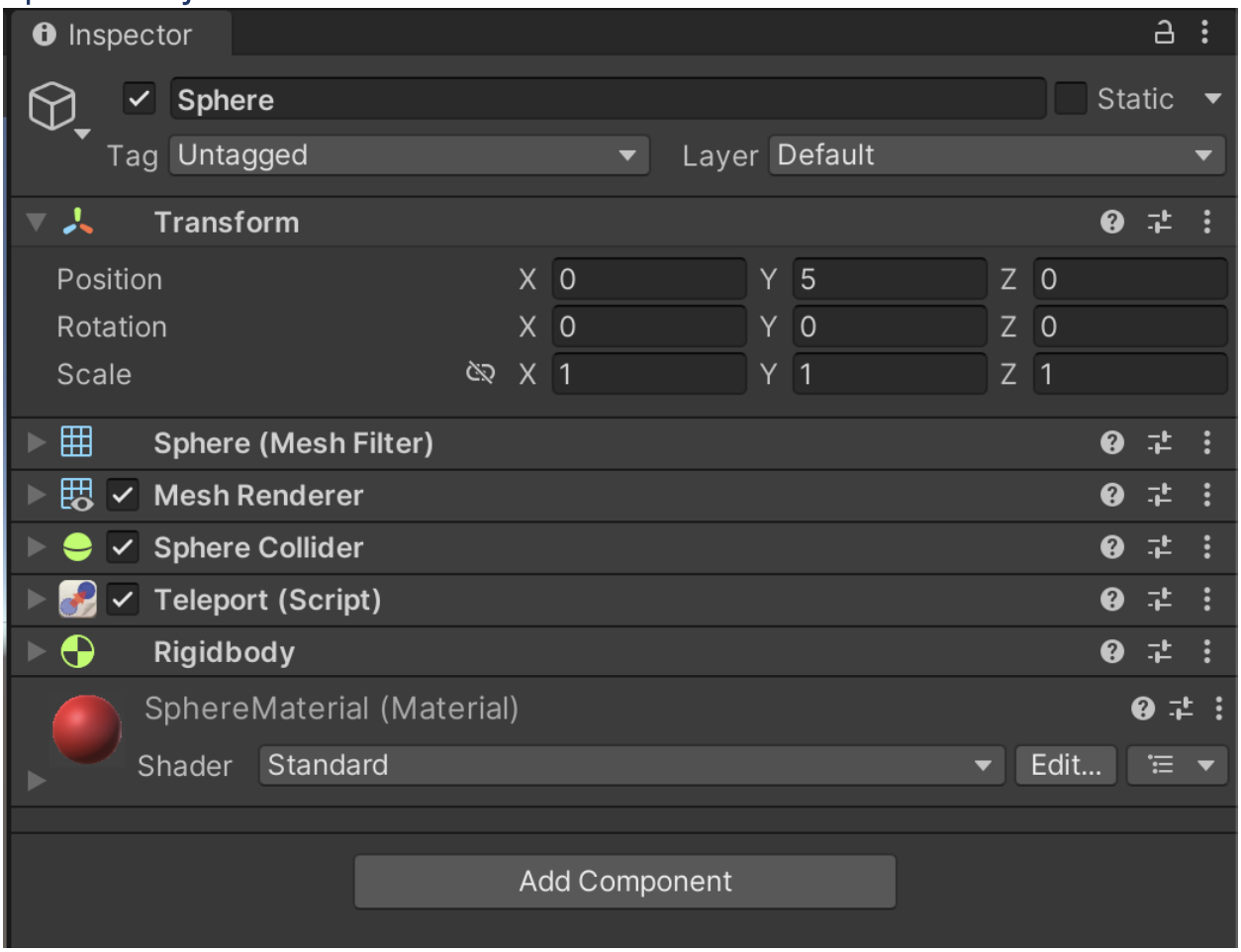
Ninjas can use any color.



Cube Object



Sphere Object



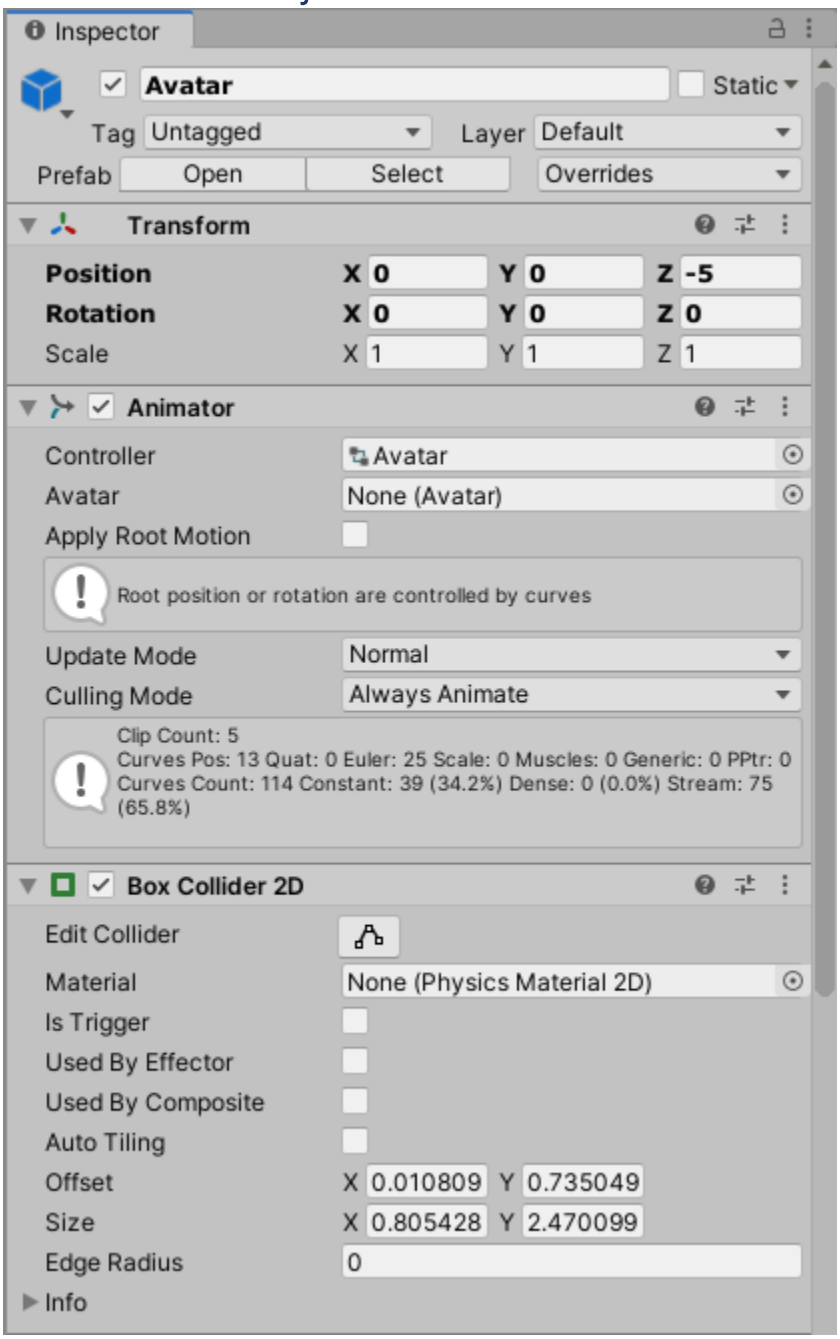
Activity Solution: Scavenger Hunt

This activity uses the **Cinemachine** Package.

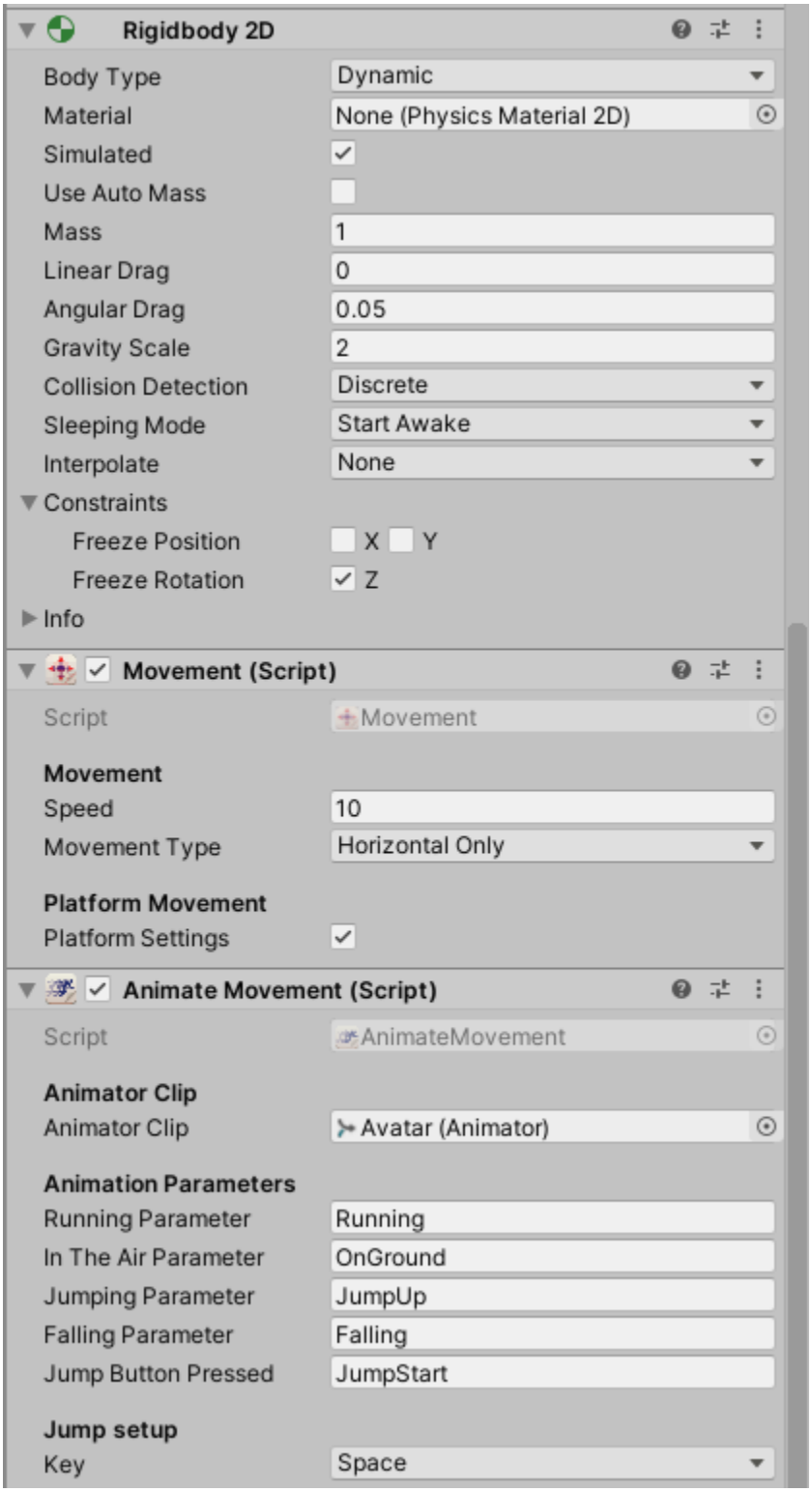
Hierarchy

The quad objects might be named based on their location in the scene.

Avatar Prefab Object

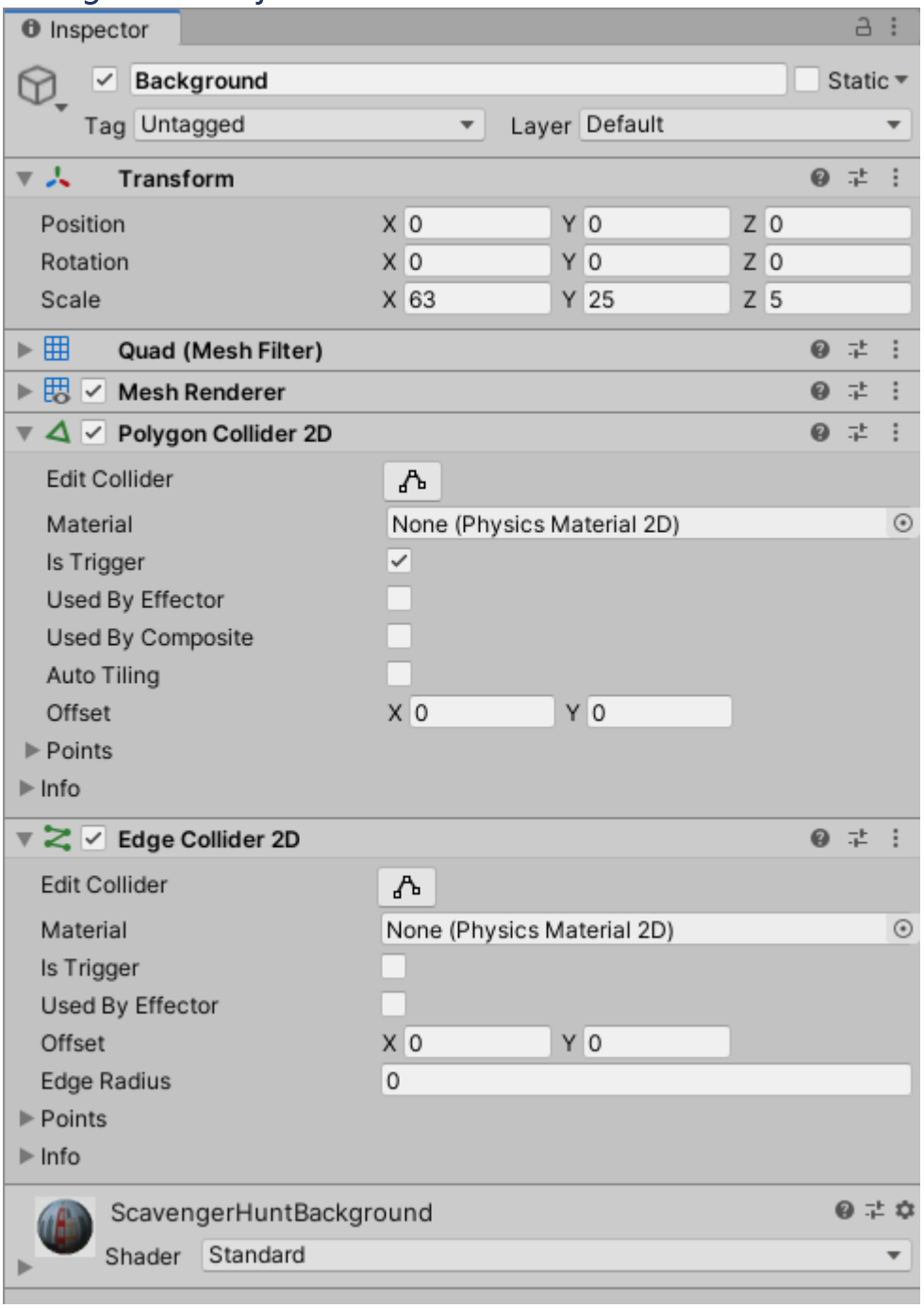


Avatar Prefab Object Continued

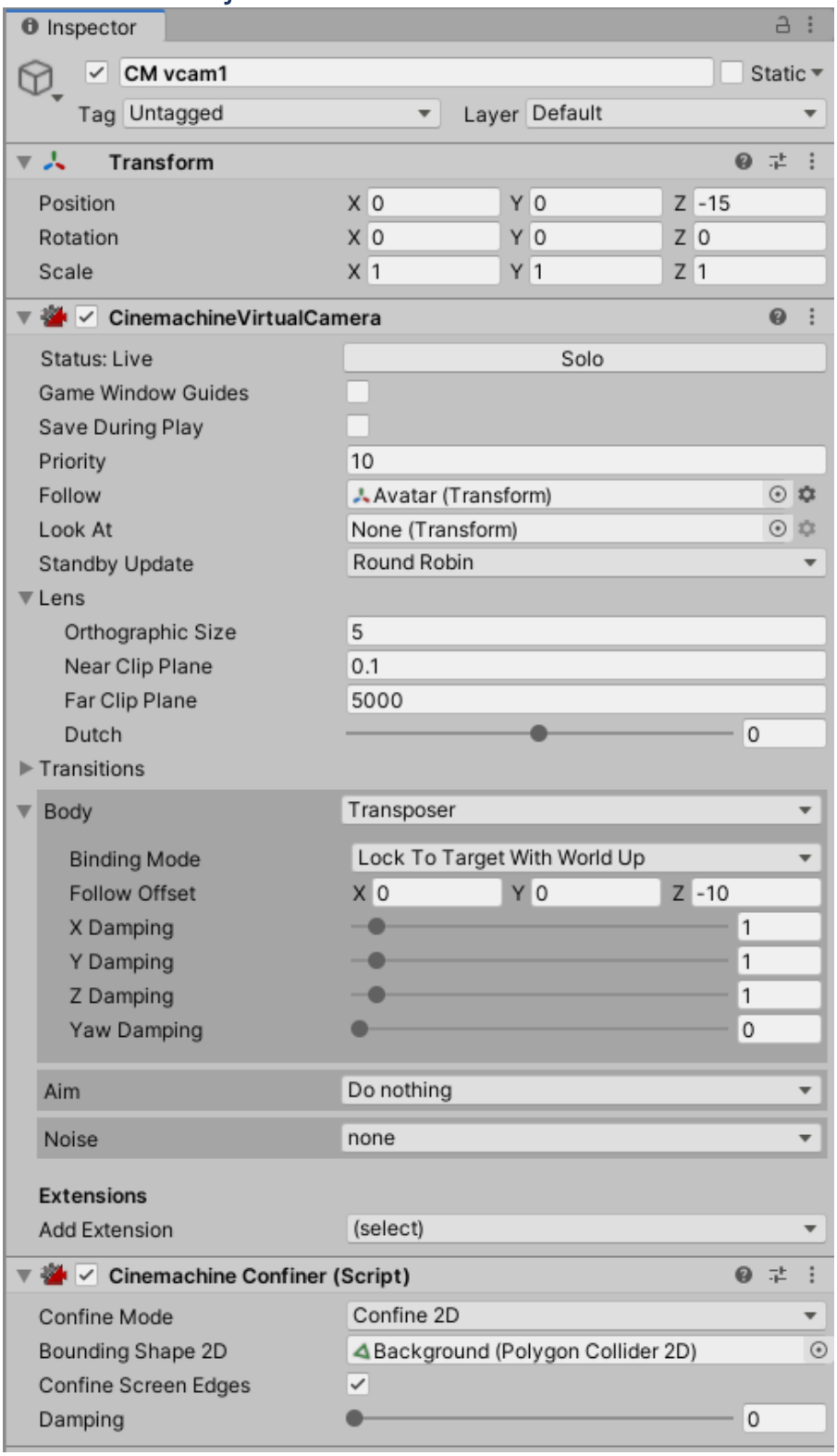


Avatar Prefab Object Continued

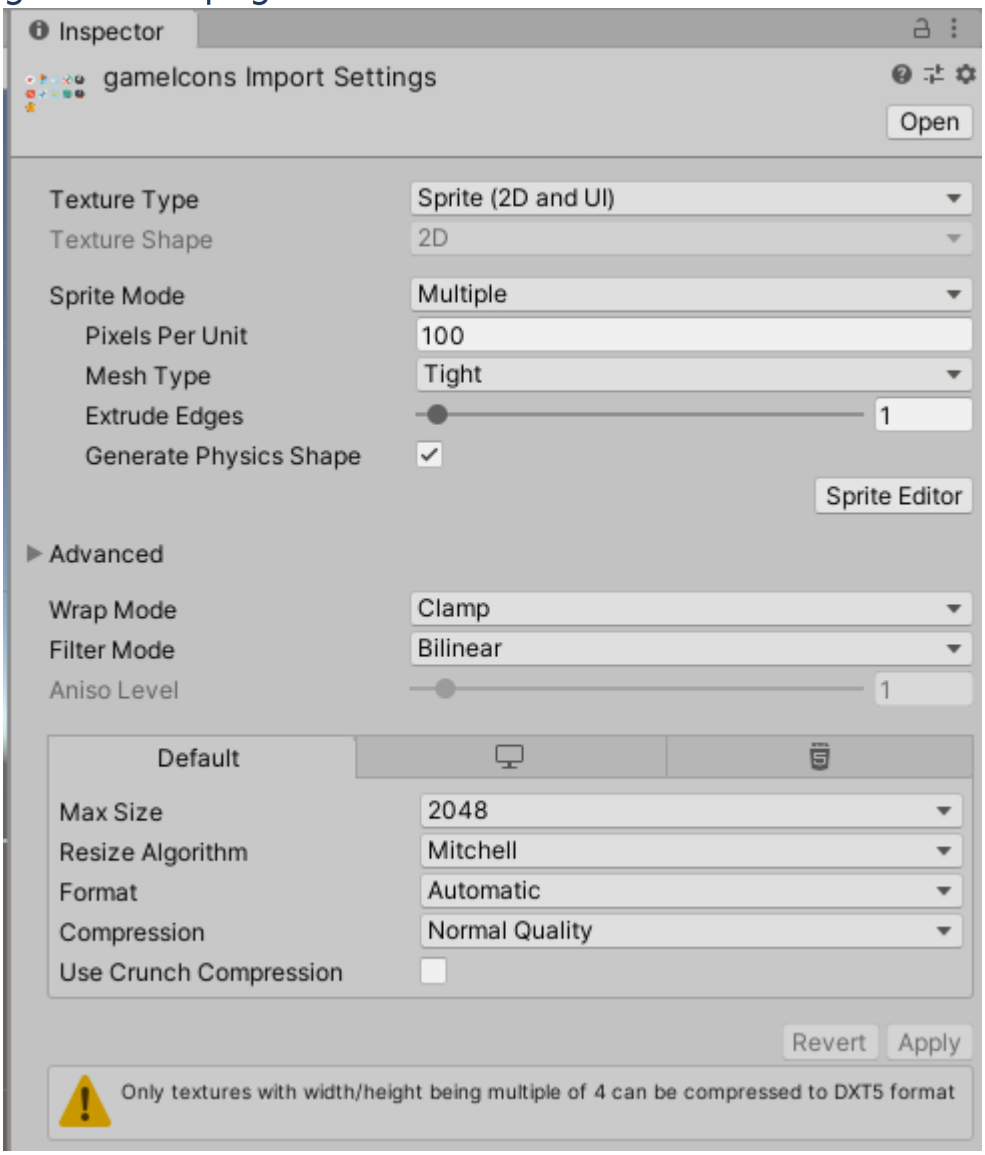
Background Object



CM vcam1 Object

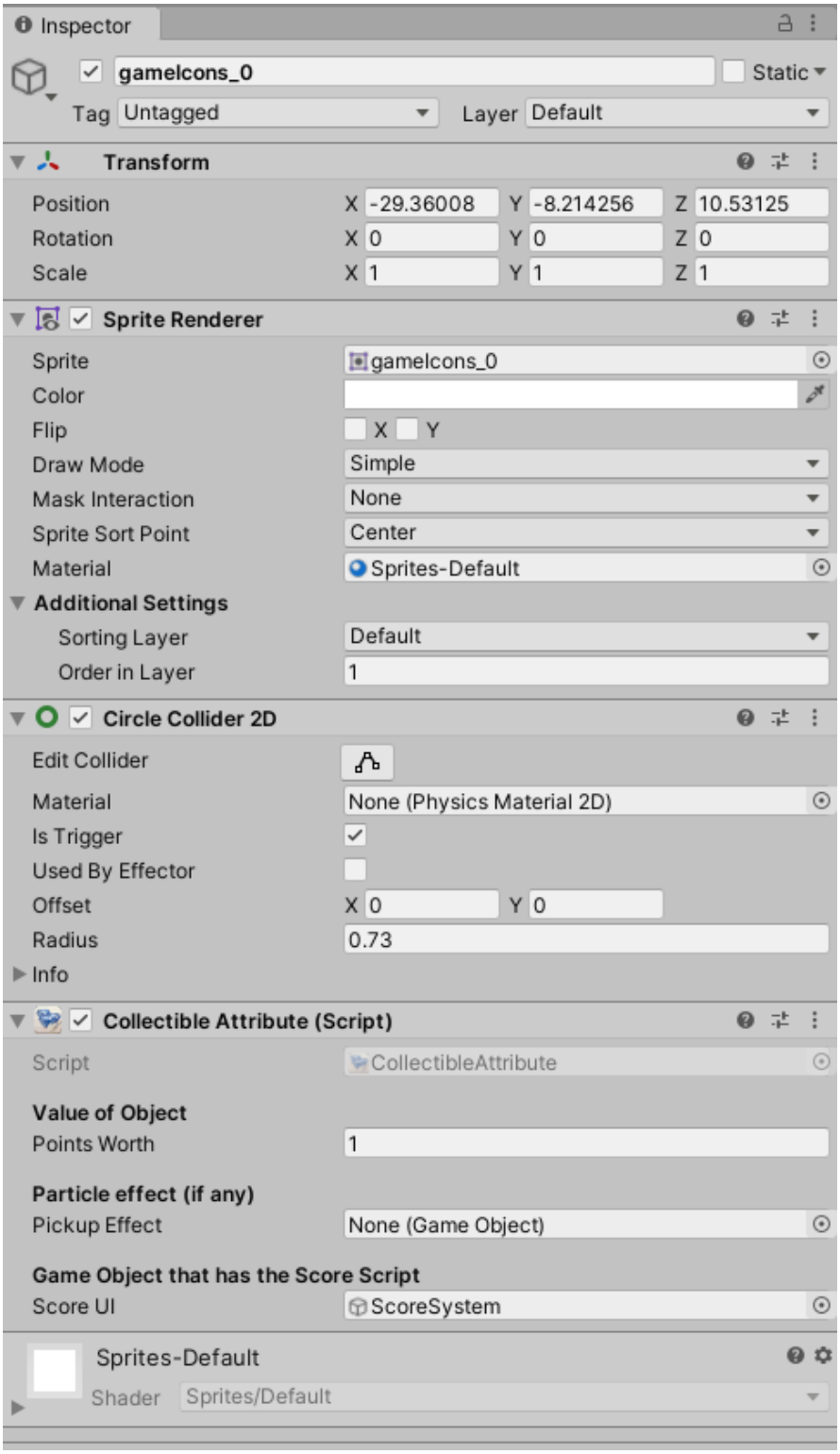


gameicons.png Asset



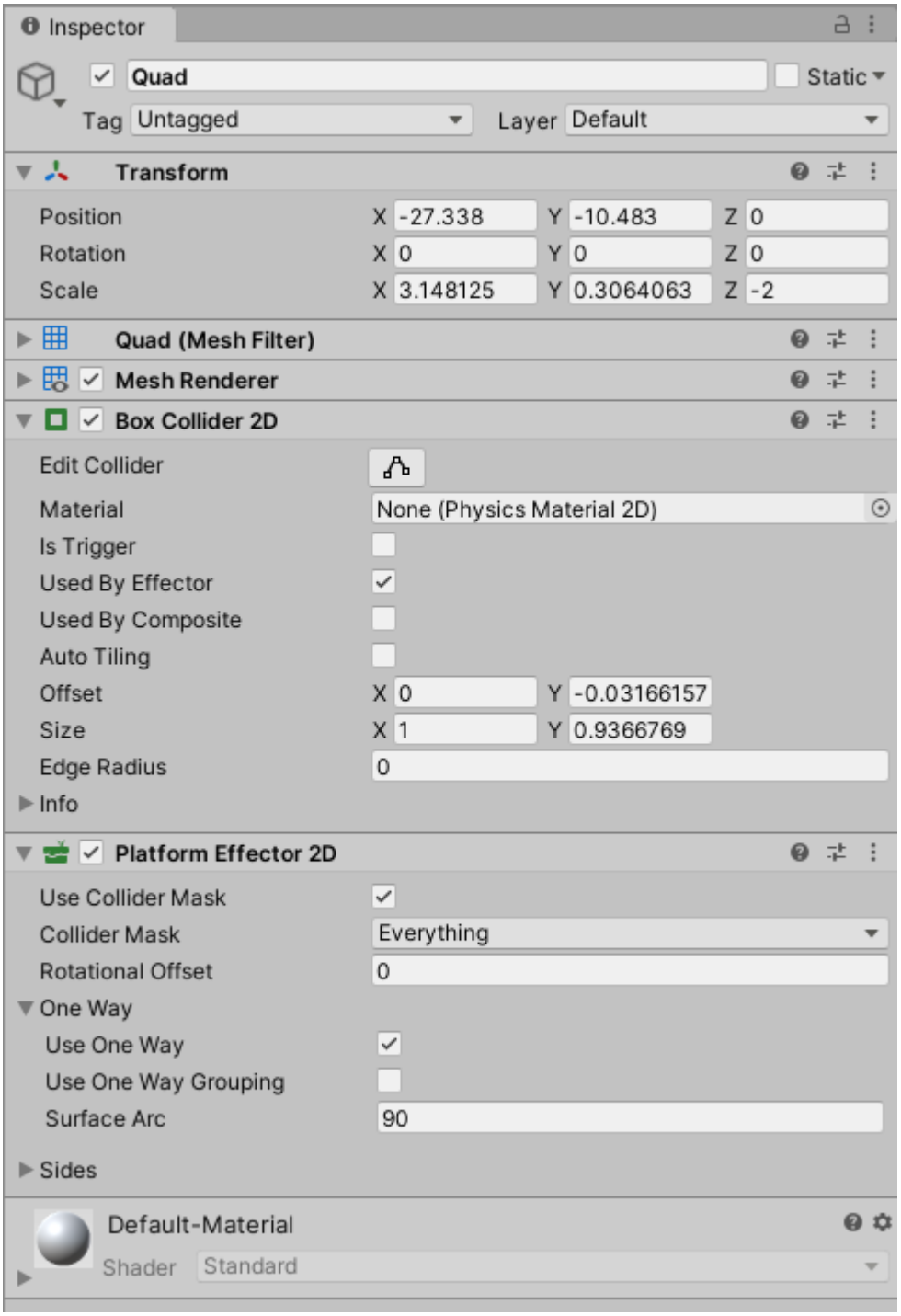
Game Icons Objects

The gamelcons objects will have different names and positions based on their locations in the scene.

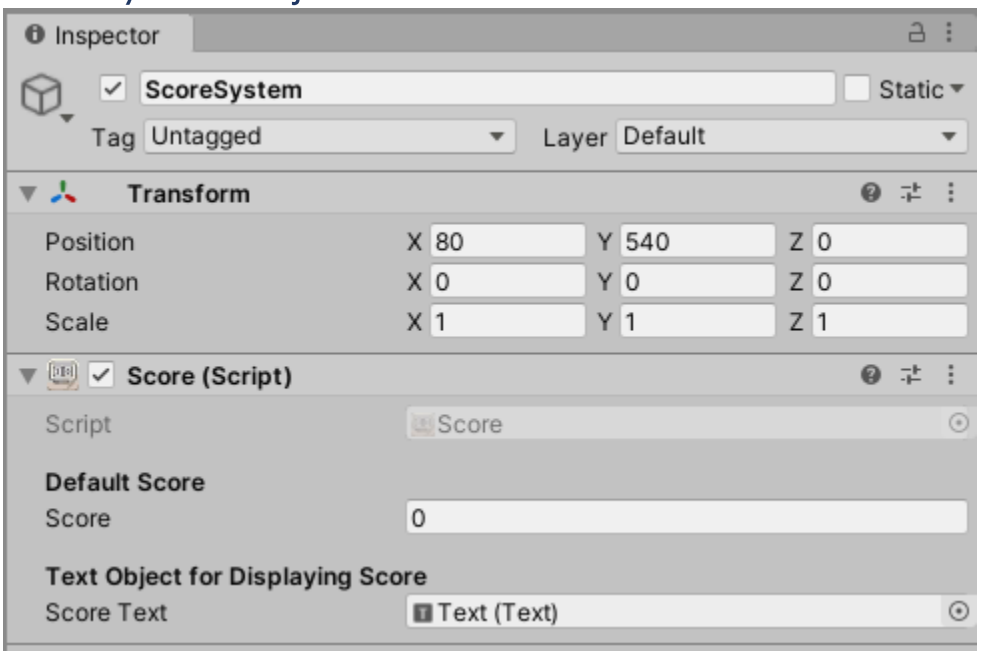


Quad objects

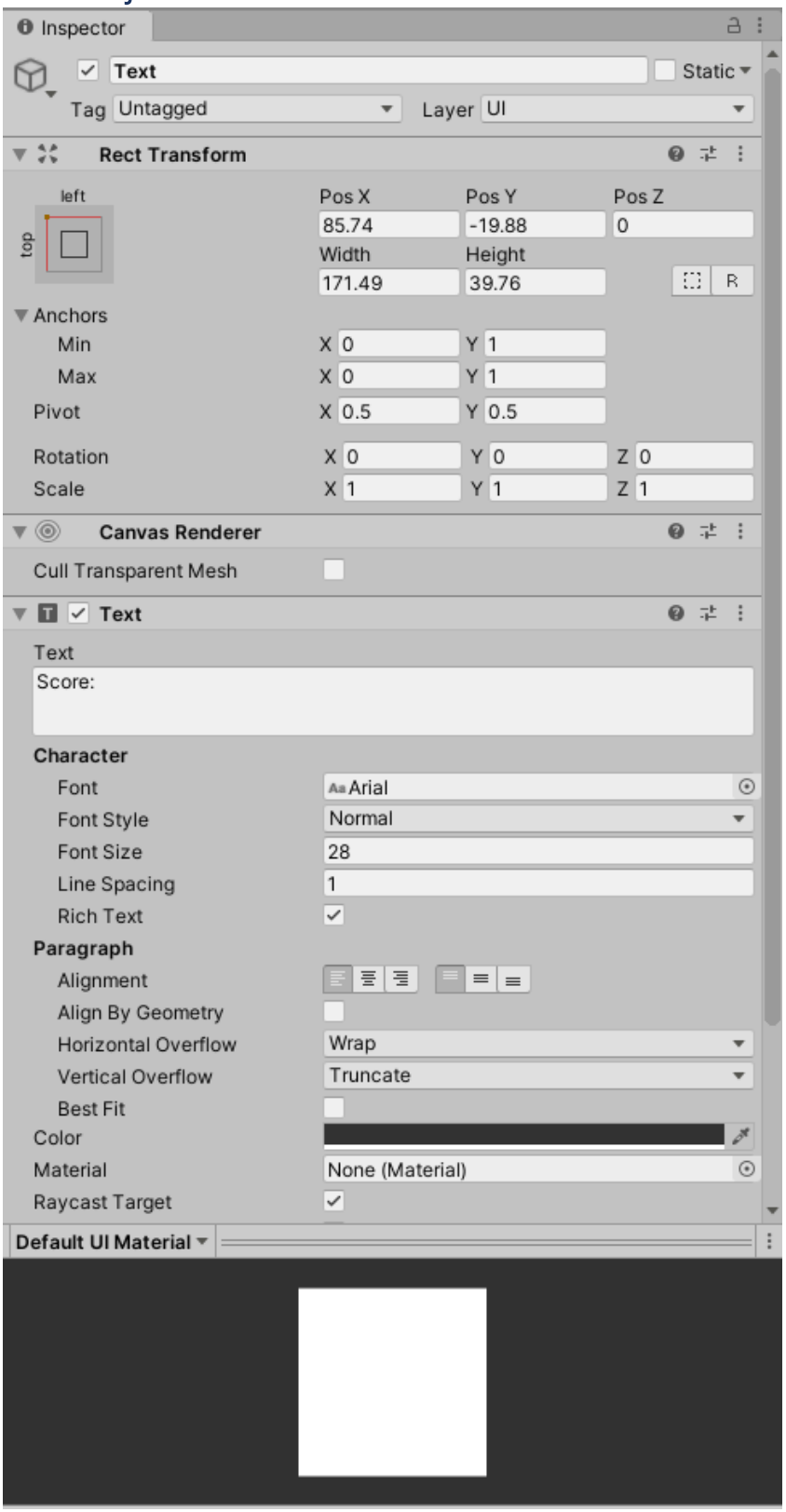
The quad objects will have a different names, positions, and scales based on their locations in the scene.



ScoreSystem Object

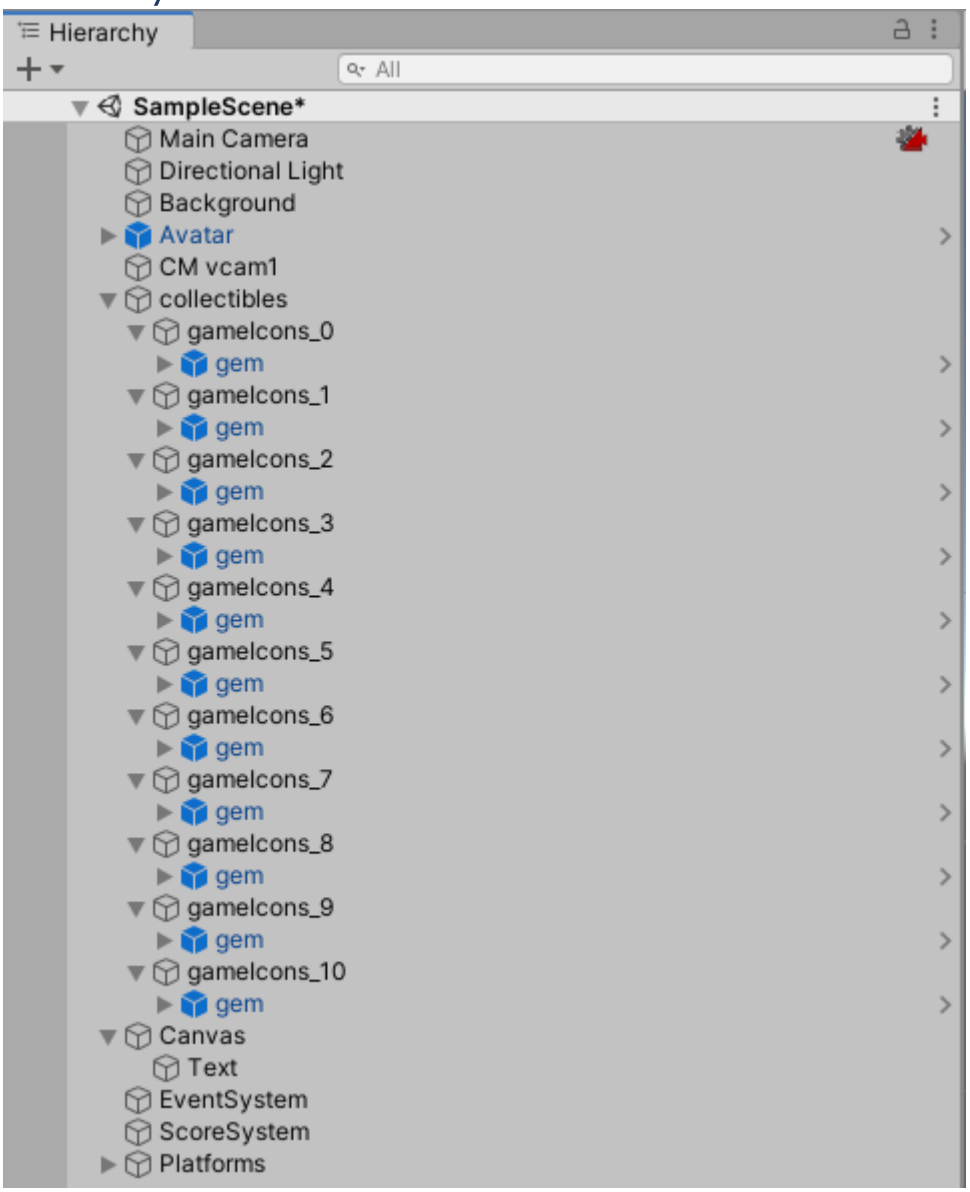


Text object

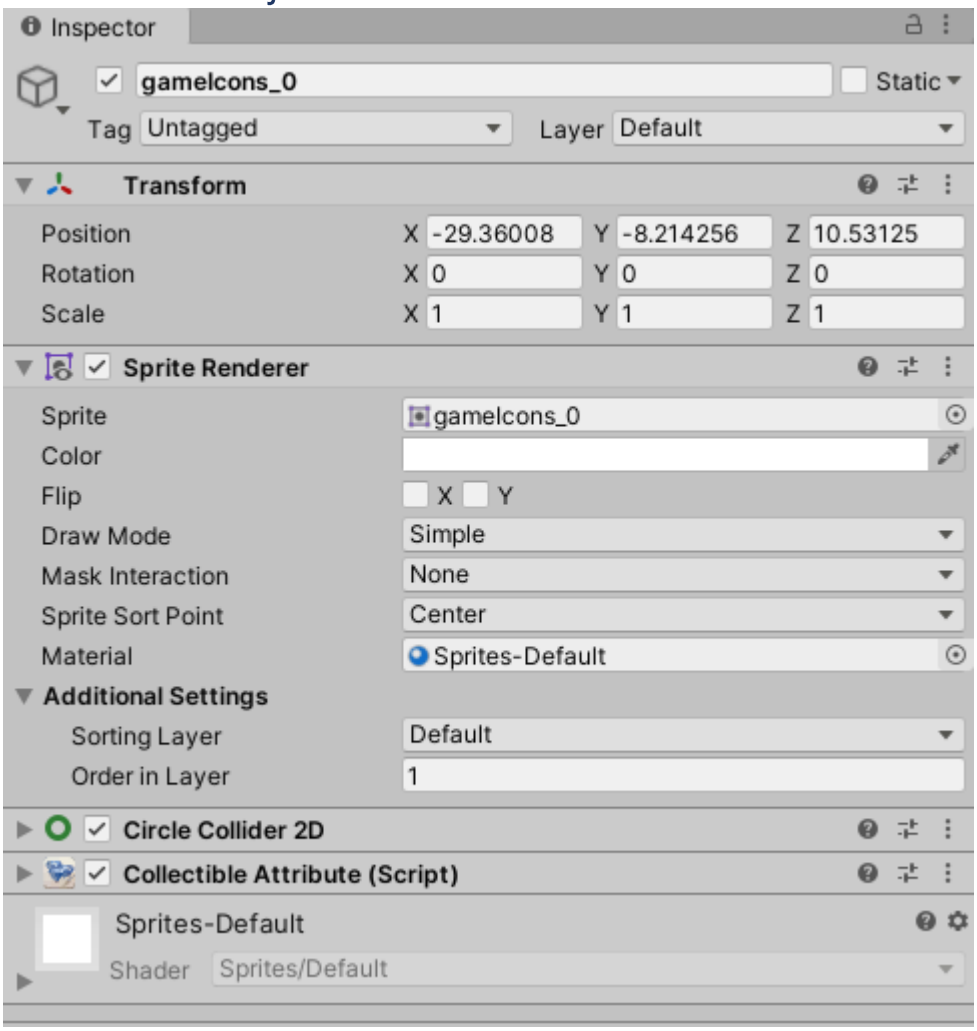


Activity Solution: Particle Hunt Prove Yourself

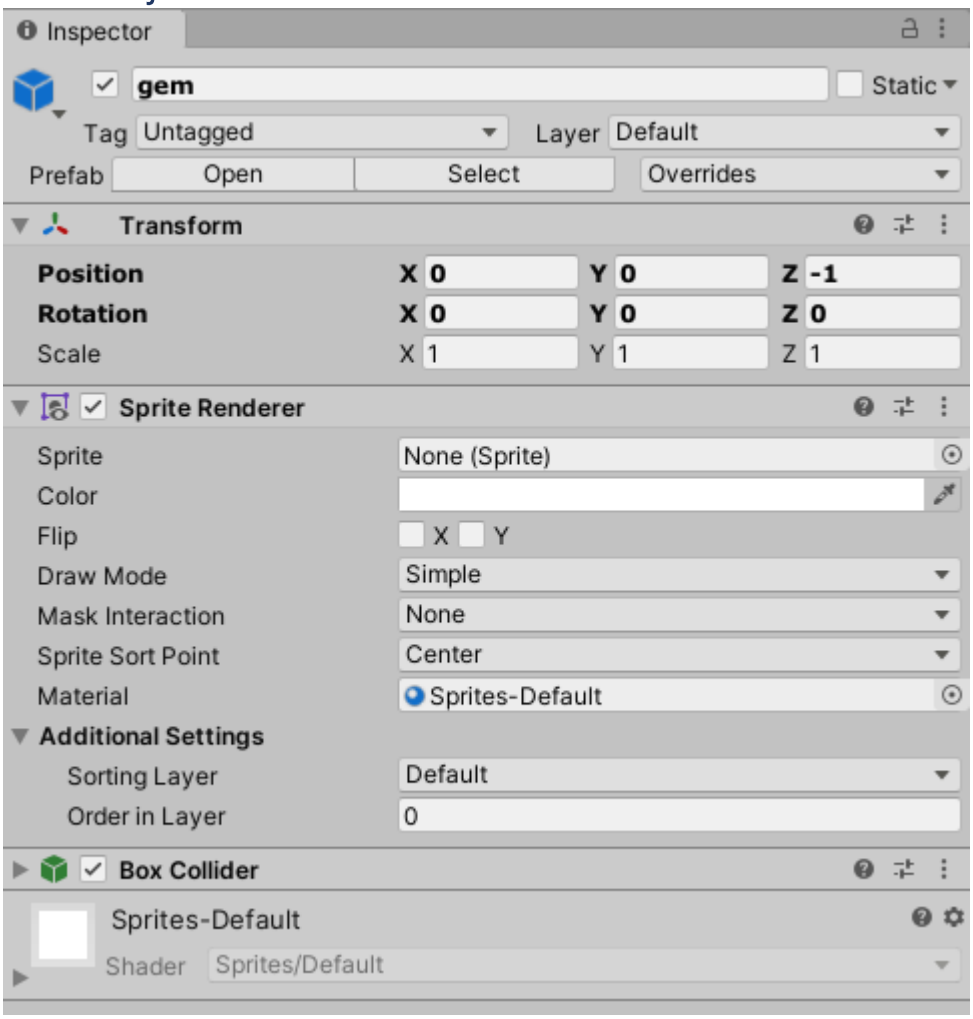
Hierarchy



Game Icons Objects

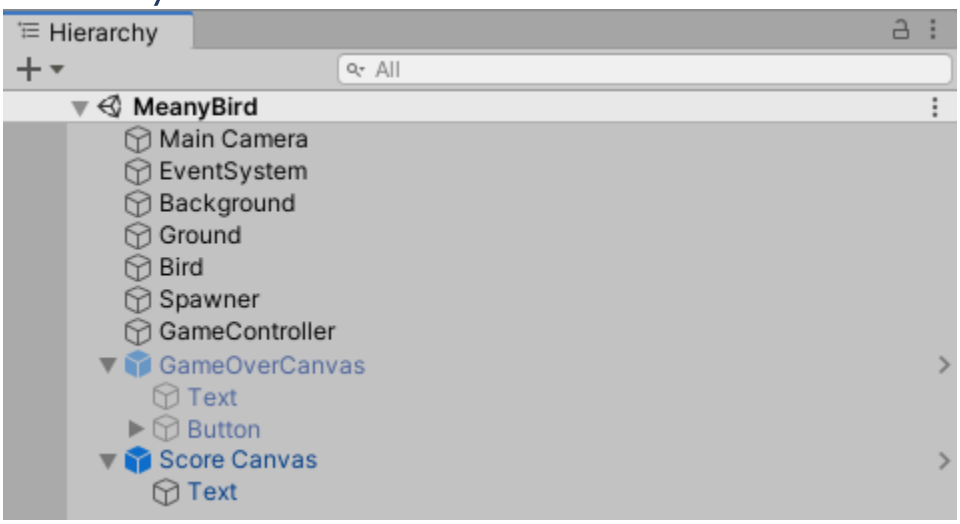


Gem Objects

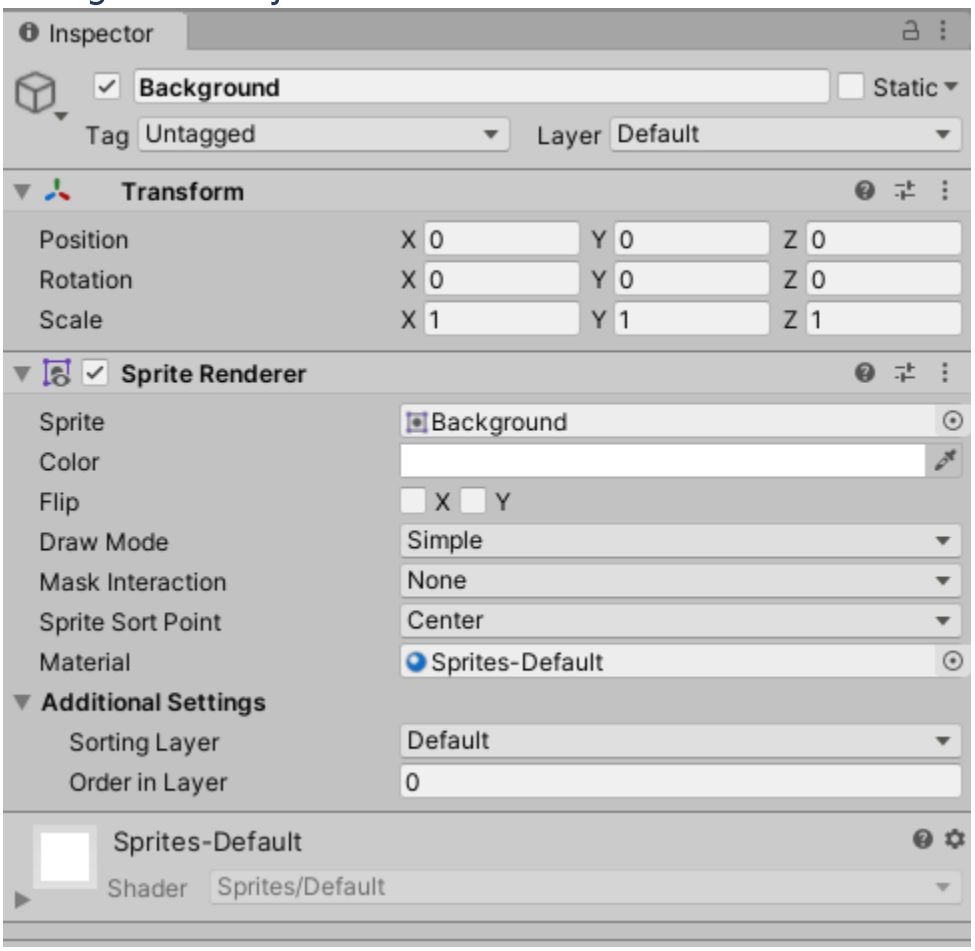


Activity Solution: Meany Bird

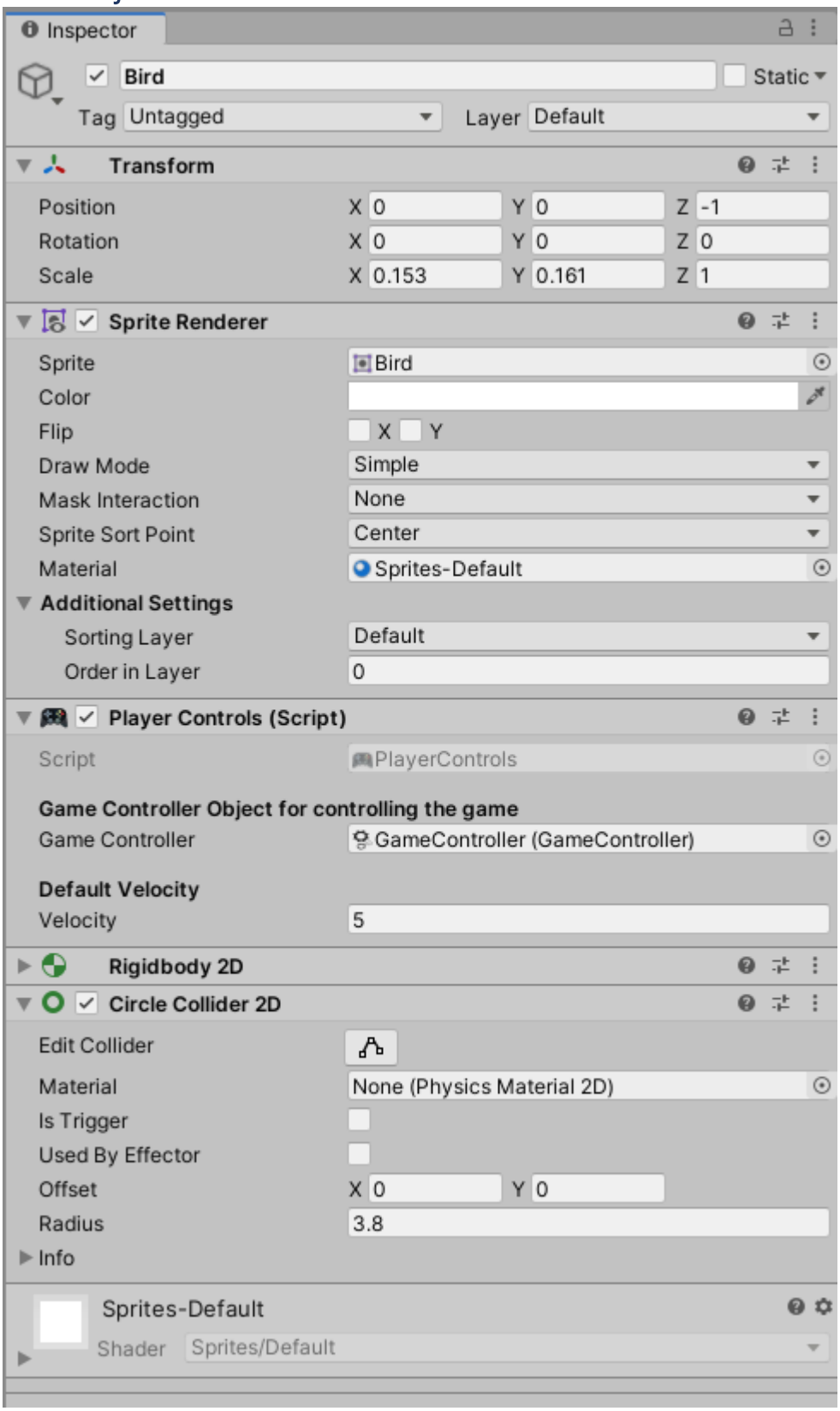
Hierarchy



Background Object



Bird Object



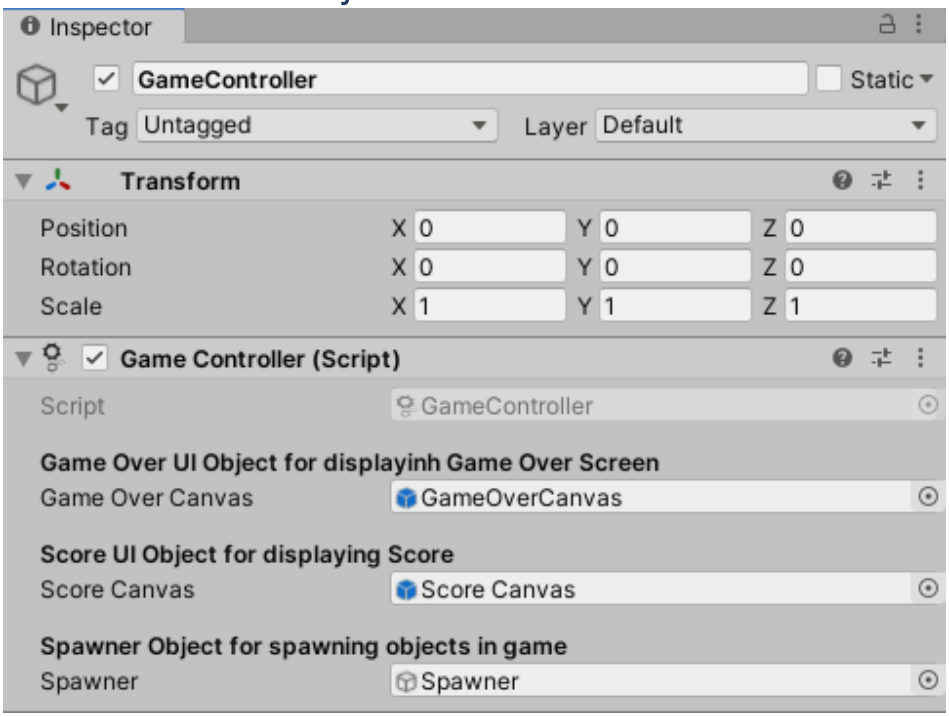
Collectibles.cs Script

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

public class Collectibles : MonoBehaviour
{
    // If an object collides with a trigger
    private void OnTriggerEnter2D(Collider2D collision)
    {
        // Add score
        Score.score++;
    }
}
```

DestroyAfterTime.cs Script

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

public class DestroyAfterTime : MonoBehaviour
{
    [Header("Default Desctrion Time")]
    public float timeToDestruction;
    // Start is called before the first frame update
    void Start()
    {
        Invoke("DestroyObject", timeToDestruction);

    }

    void DestroyObject()
    {
        Destroy(gameObject);
    }
}
```

GameController Object



GameController.cs Script

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

public class GameController : MonoBehaviour
{
    [Header("Game Over UI Object for displayinh Game Over Screen")]
    public GameObject gameOverCanvas;
    [Header("Score UI Object for displaying Score")]
    public GameObject scoreCanvas;
    [Header("Spawner Object for spawning objects in game")]
    public GameObject spawner;

    // Start is called before the first frame update
    void Start()
    {
        Time.timeScale = 1;
        scoreCanvas.SetActive(true);
        gameOverCanvas.SetActive(false);
        spawner.SetActive(true);
    }

    public void GameOver()
    {
        gameOverCanvas.SetActive(true);
        spawner.SetActive(false);
        Time.timeScale = 0;
    }

}
```

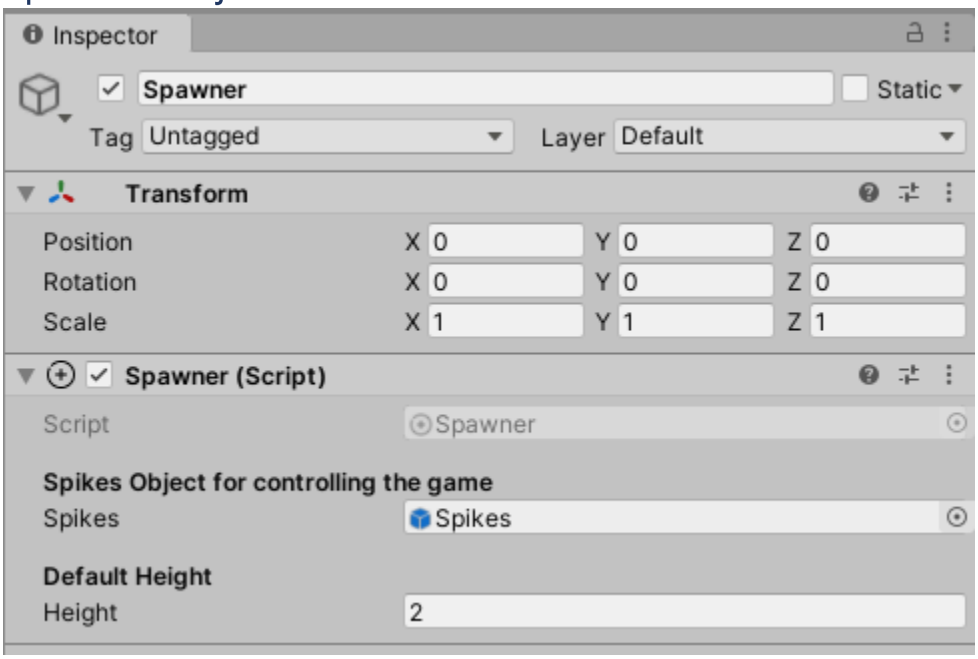
Move.cs Script

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

public class Move : MonoBehaviour
{
    [Header("Default Speed")]
    public float speed;

    // Update is called once per frame
    void Update()
    {
        transform.position += Vector3.left * speed * Time.deltaTime;
    }
}
```

PlayerControls.cs Script

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

public class PlayerControls : MonoBehaviour
{
    [Header("Game Controller Object for controlling the game")]
    public GameController gameController;
    [Header("Default Velocity")]
    public float velocity = 1;
    private Rigidbody2D rb;
    private float objectHeight;

    // Start is called before the first frame update
    void Start()
    {
        gameController = GetComponent<GameController>();
        Time.timeScale = 1;
        rb = GetComponent<Rigidbody2D>();
        objectHeight = transform.GetComponent<SpriteRenderer>().bounds.size.y / 2;

    }

    // Update is called once per frame
    void Update()
    {
        if(Input.GetMouseButtonDown(0)){
            rb.velocity = Vector2.up*velocity;
        }

    }
    private void OnCollisionEnter2D(Collision2D collision)
    {
        if (collision.gameObject.tag == "HighSpike"
            || collision.gameObject.tag == "LowSpike"
            || collision.gameObject.tag == "Ground") {
            GameObject.Find("GameController").GetComponent<GameController>().GameOver();
        }
    }
}
```

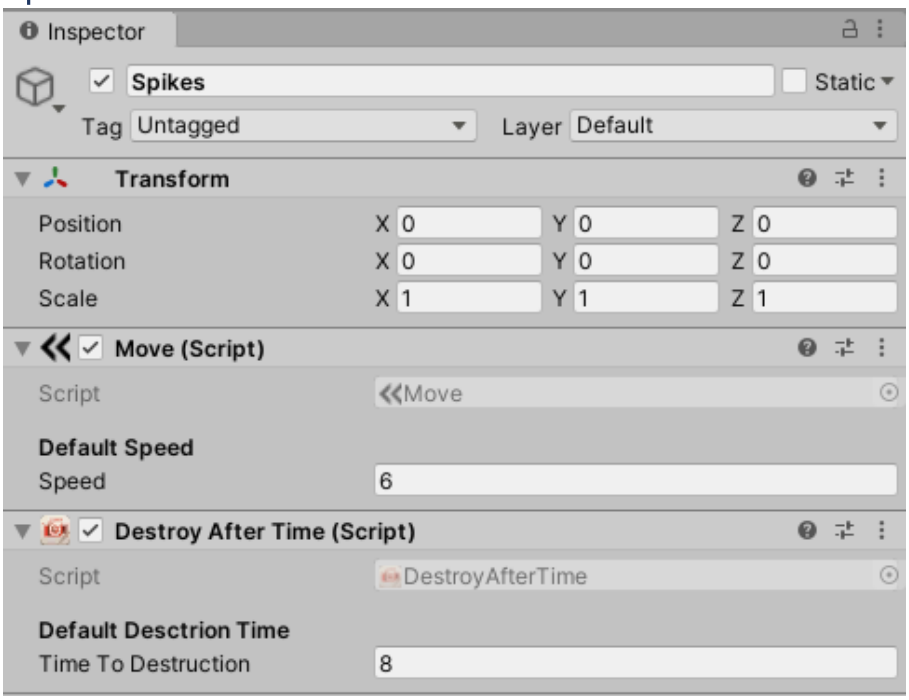
Spawner Object



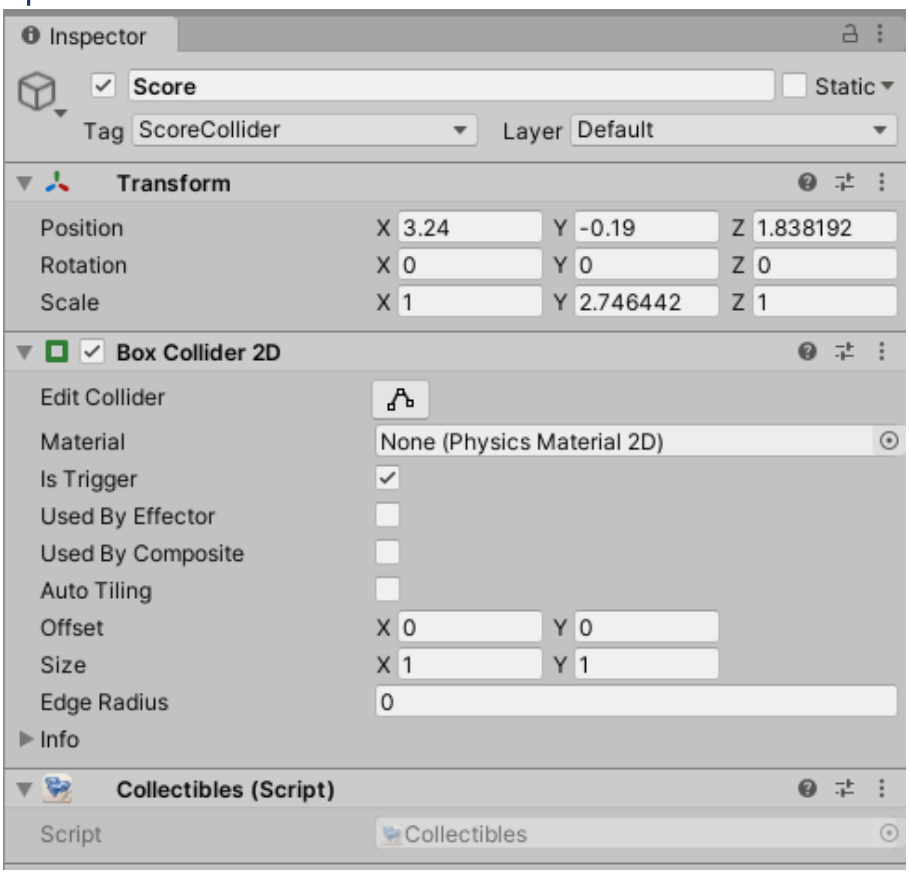
Spawner.cs Script

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

public class Spawner : MonoBehaviour
{
    [Header("Spikes Object for controlling the game")]
    public GameObject spikes;
    [Header("Default Height")]
    public float height;
    // Start is called before the first frame update
    void Start()
    {
        InvokeRepeating("InstantiateObjects", 1f, 4f);

    }

    // Update is called once per frame
    void Update()
    {
        transform.position = new Vector3(10, Random.Range(-height, height), 0);

    }

    void InstantiateObjects()
    {
        Instantiate(spikes, transform.position, transform.rotation);
    }
}
```

Spikes Prefab

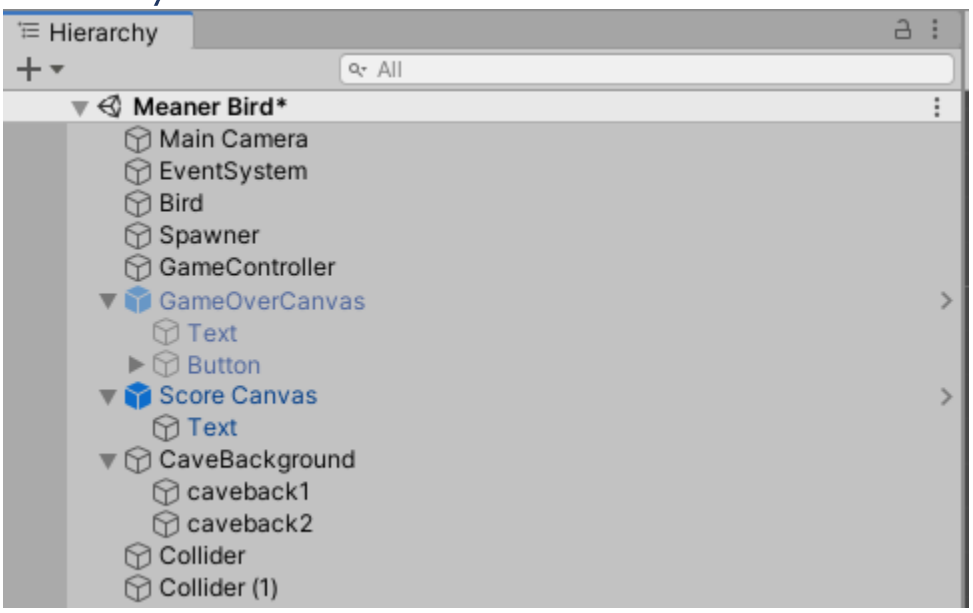


Spikes > Score Prefab

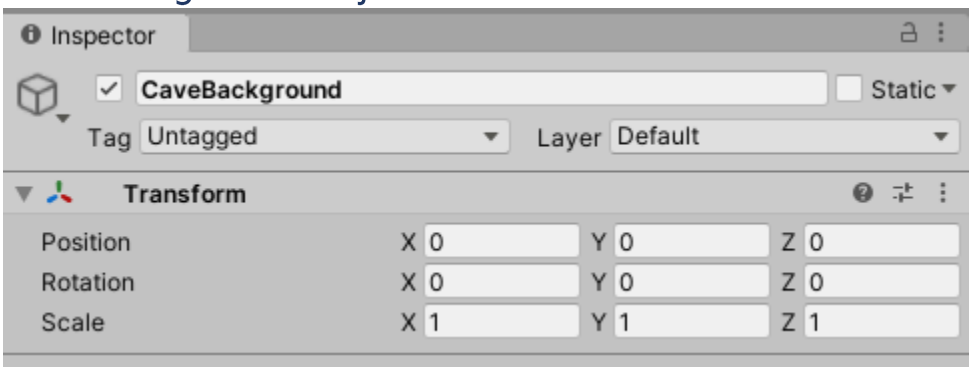


Activity Solution: Meaner Bird Prove Yourself

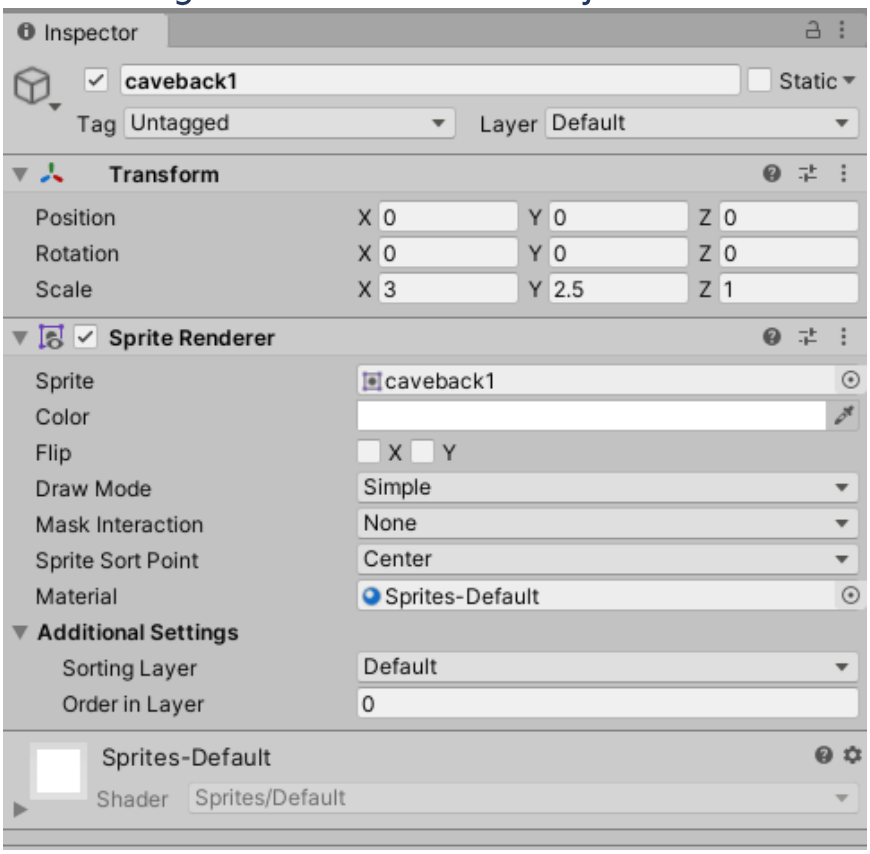
Hierarchy



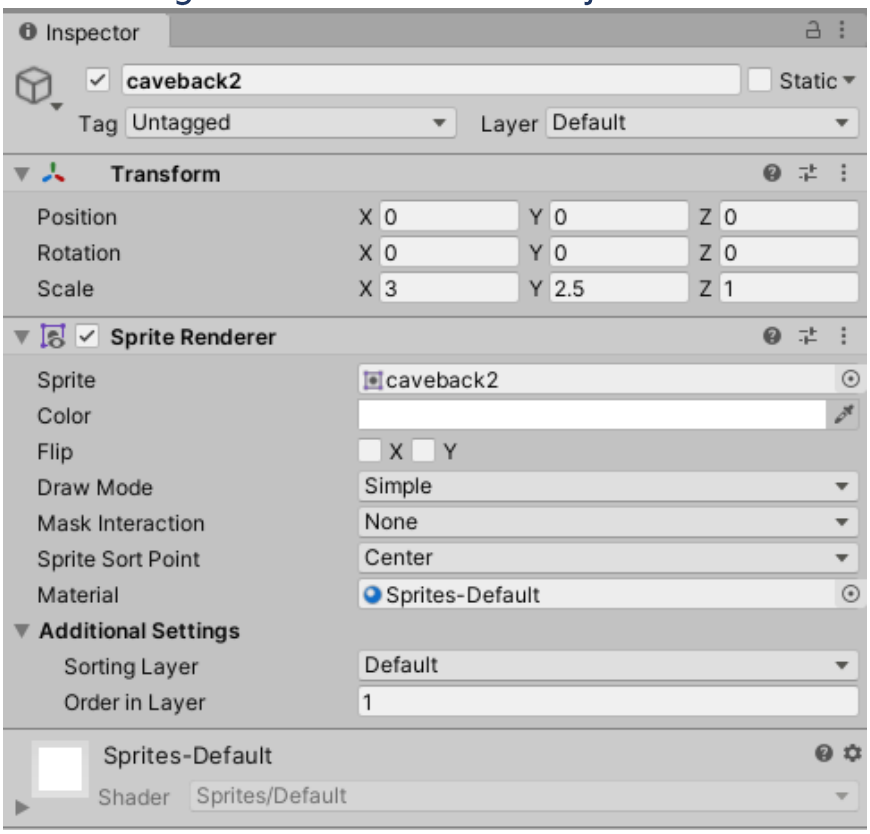
CaveBackground Object



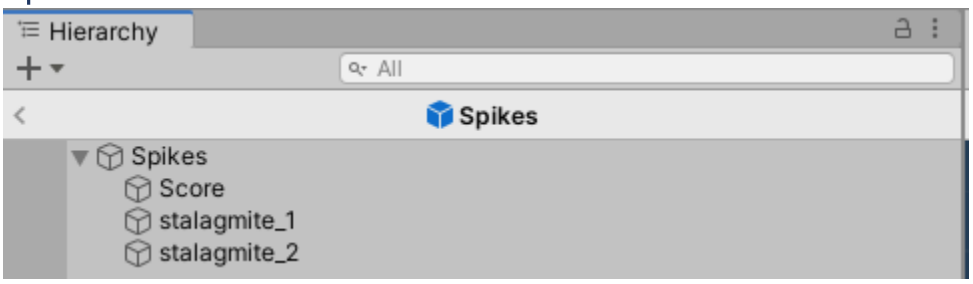
CaveBackground > caveback1 Object



CaveBackground > caveback2 Object

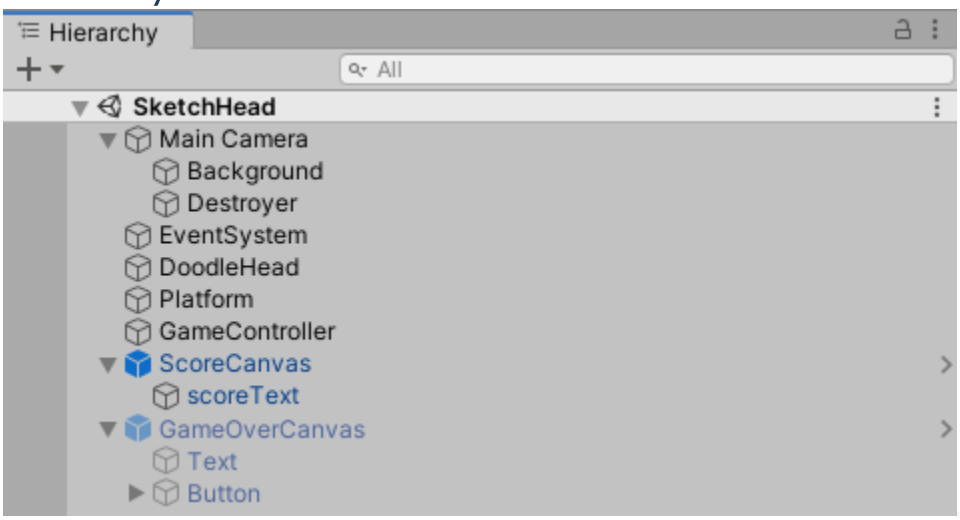


Spikes Prefab



Activity Solution: Sketch Head

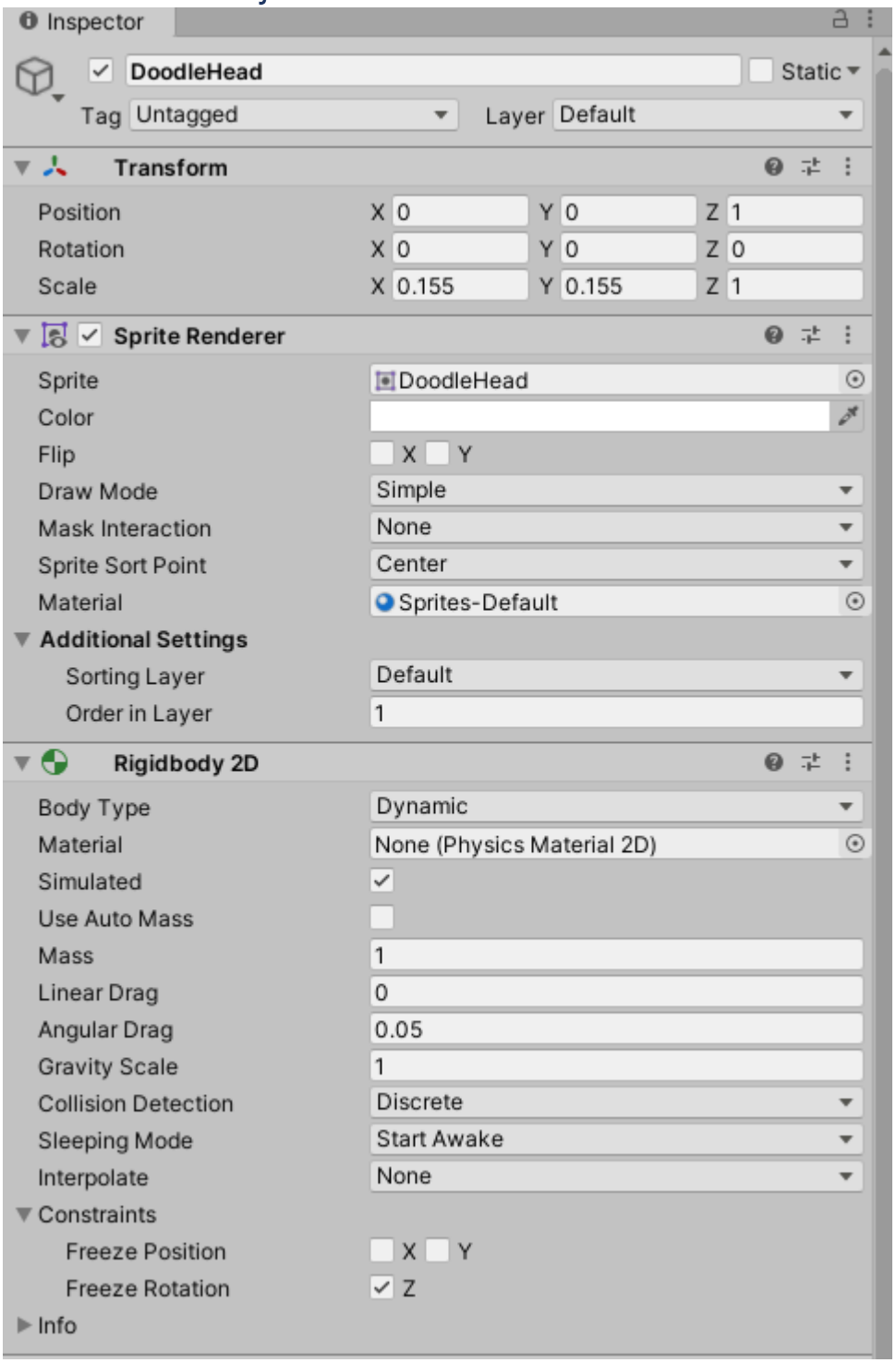
Hierarchy



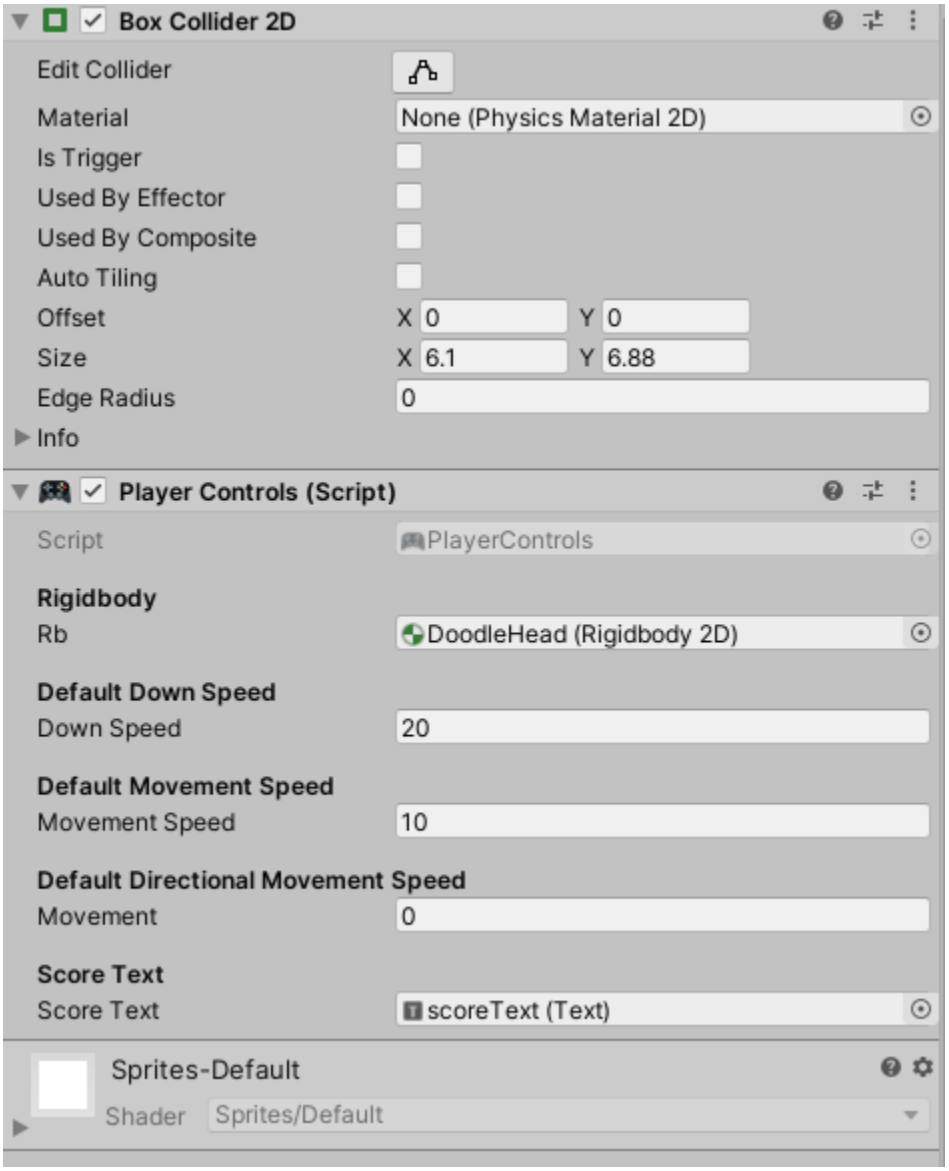
Destroyer.cs Script

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

public class Destroyer : MonoBehaviour
{
    public void OnTriggerEnter2D(Collider2D collision)
    {
        GameObject.Find("DoodleHead").SetActive(false);
        GameObject.Find("GameController").GetComponent<GameController>().GameOver();
    }
}
```

DoodleHead Object



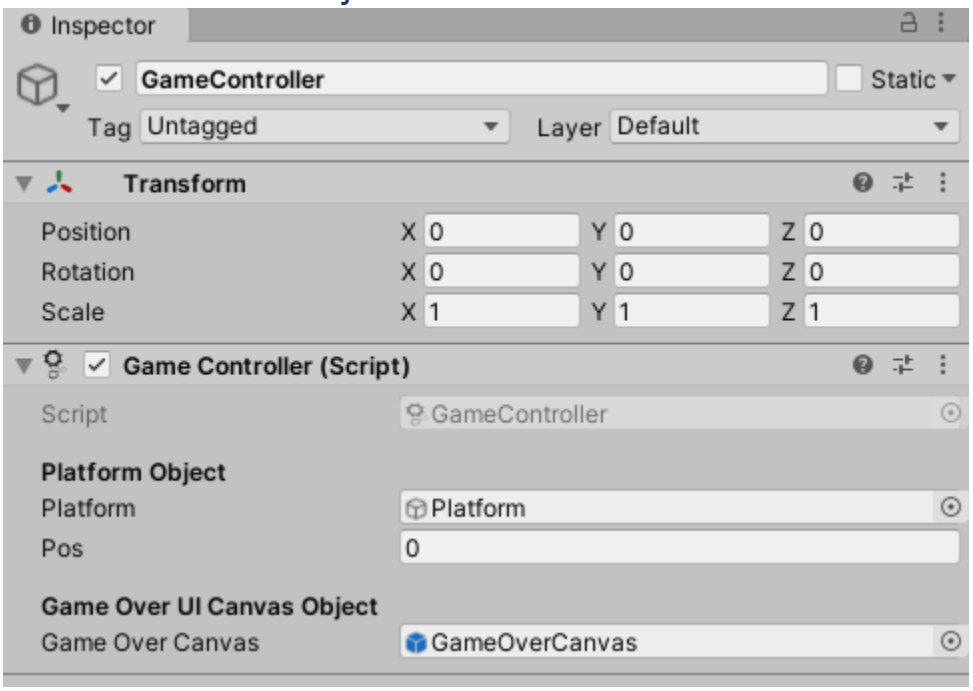
DoodleHead object continued:



CameraFollow.cs Script

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

public class CameraFollow : MonoBehaviour
{
    [Header("Target Object")]
    public Transform target;

    // Update is called once per frame
    private void Update()
    {
        if (target.position.y > transform.position.y)
        {
            transform.position = new Vector3(
                target.transform.position.x,
                target.transform.position.y,
                transform.position.z);
        }
    }
}
```

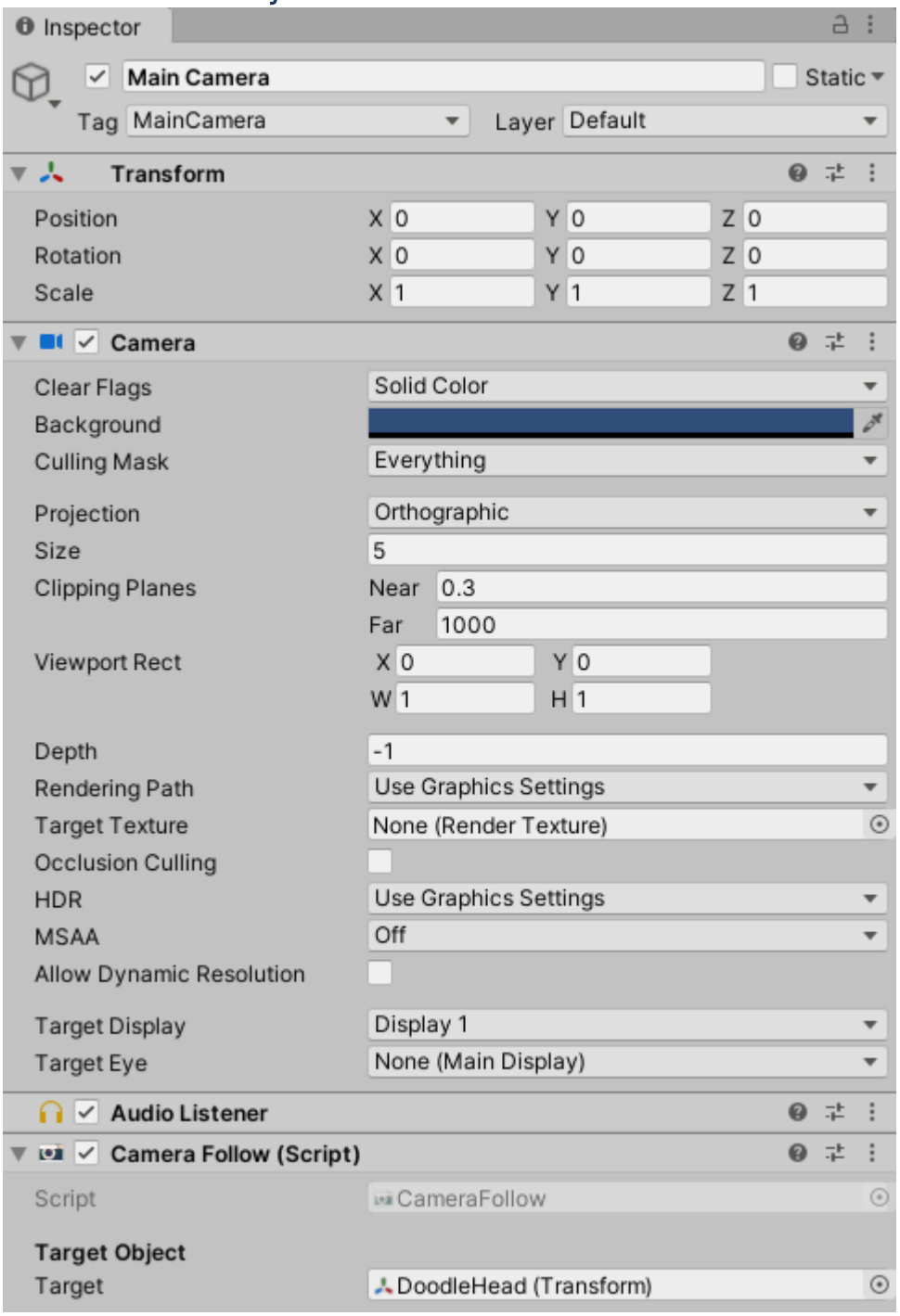
GameController Object



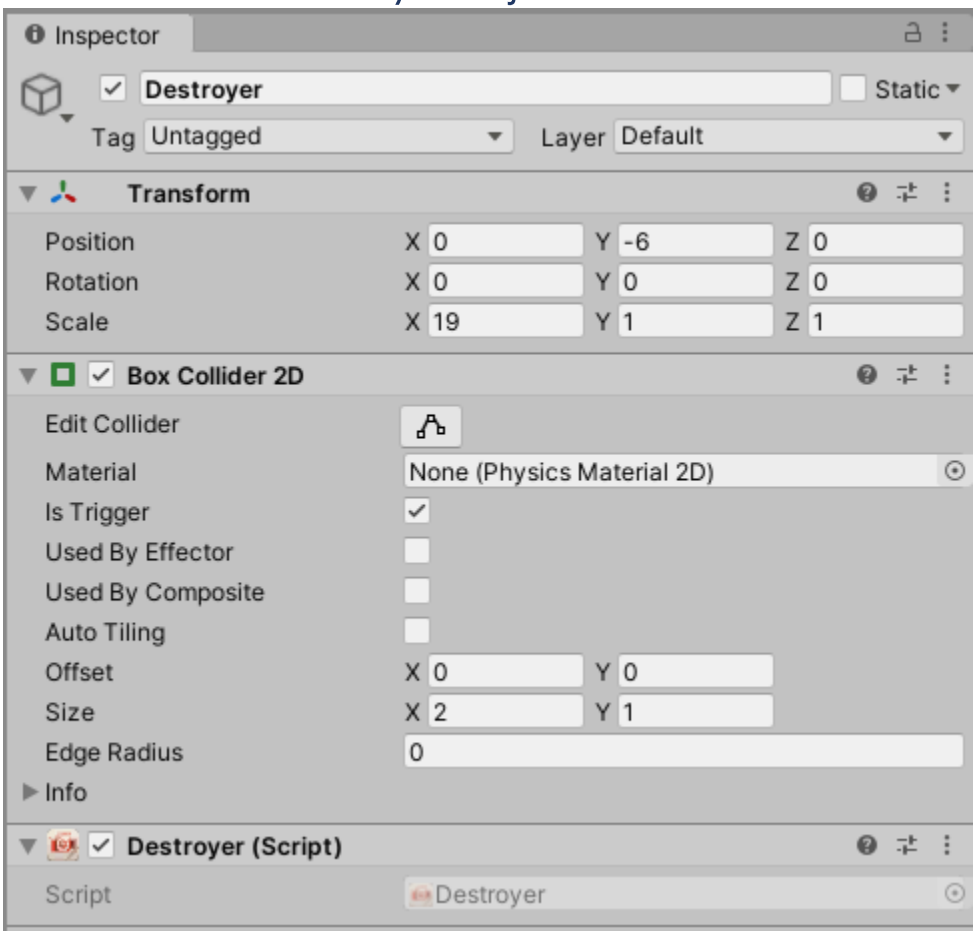
GameController.cs Script

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

public class GameController : MonoBehaviour
{
    [Header("Platform Object")]
    public GameObject platform;
    public float pos = 0;
    [Header("Game Over UI Canvas Object")]
    public GameObject gameOverCanvas;

    void Start()
    {
        for (int i = 0; i < 1000; i++)
        {
            SpawnPlatforms();
        }
    }

    void SpawnPlatforms()
    {
        Instantiate(platform, new Vector3(
            Random.value * 10 - 5f, pos, 0.5f),
            Quaternion.identity);
        pos += 5f;
    }

    public void GameOver()
    {
        gameOverCanvas.SetActive(true);
    }
}
```

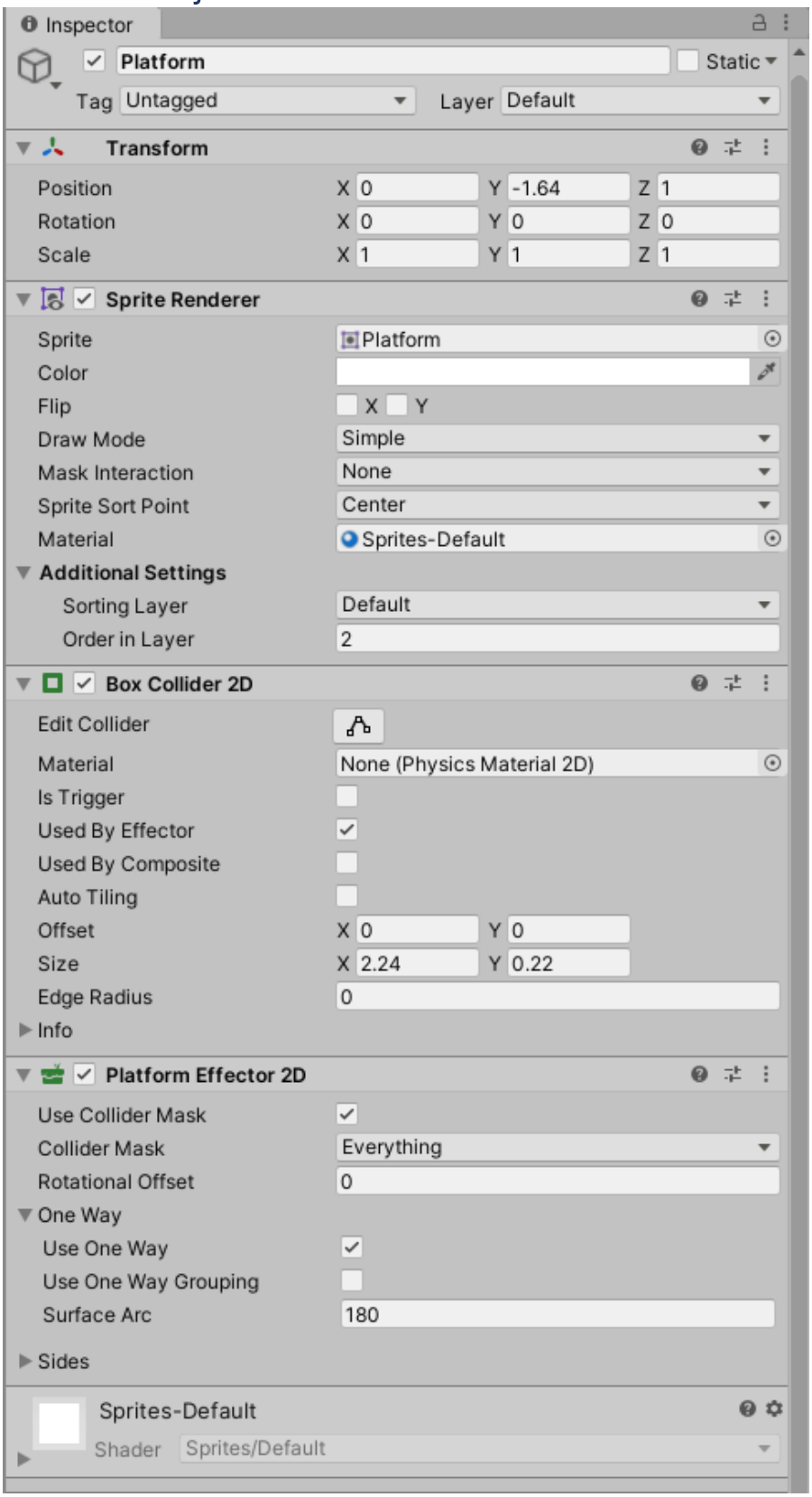
Main Camera Object



Main Camera > Destroyer Object



Platform Object

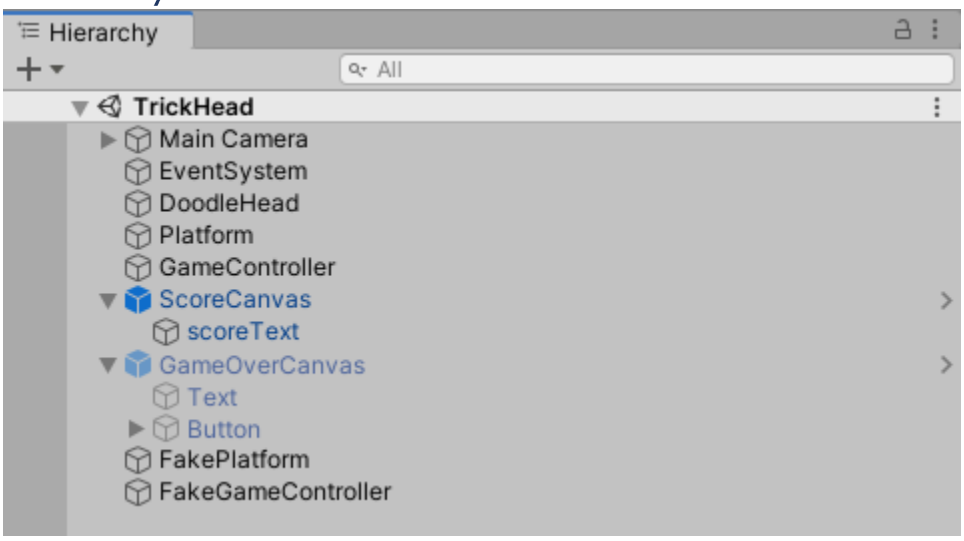


PlayerControls.cs Script

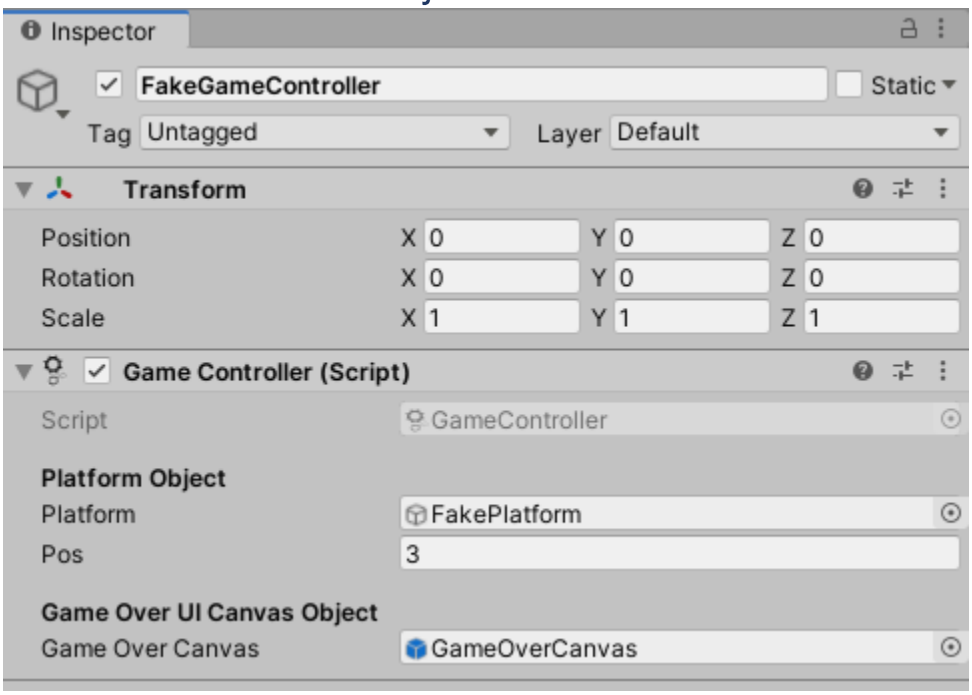
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class PlayerControls : MonoBehaviour
{
    [Header("Rigidbody")]
    public Rigidbody2D rb;
    [Header("Default Down Speed")]
    public float downSpeed = 20f;
    [Header("Default Movement Speed")]
    public float movementSpeed = 10f;
    [Header("Default Directional Movement Speed")]
    public float movement = 0f;
    [Header("Score Text")]
    public Text scoreText;
    private float topScore = 0.0f;

    void Start()
    {
        rb = GetComponent<Rigidbody2D>();
    }

    void Update()
    {
        movement = Input.GetAxis("Horizontal") * movementSpeed;
        if (movement < 0)
        {
            this.GetComponent<SpriteRenderer>().flipX = false;
        }
        else
        {
            this.GetComponent<SpriteRenderer>().flipX = true;
        }
        if (rb.velocity.y > 0 && transform.position.y > topScore)
        {
            topScore = transform.position.y;
        }
        scoreText.text = "Score: " + Mathf.Round(topScore).ToString();
    }

    void FixedUpdate()
    {
        Vector2 velocity = rb.velocity;
        velocity.x = movement;
        rb.velocity = velocity;
    }

    private void OnCollisionEnter2D(Collision2D collision)
    {
        rb.velocity = new Vector3(rb.velocity.x, downSpeed, 0);
    }
}
```

Activity Solution: Trick Head Prove Yourself

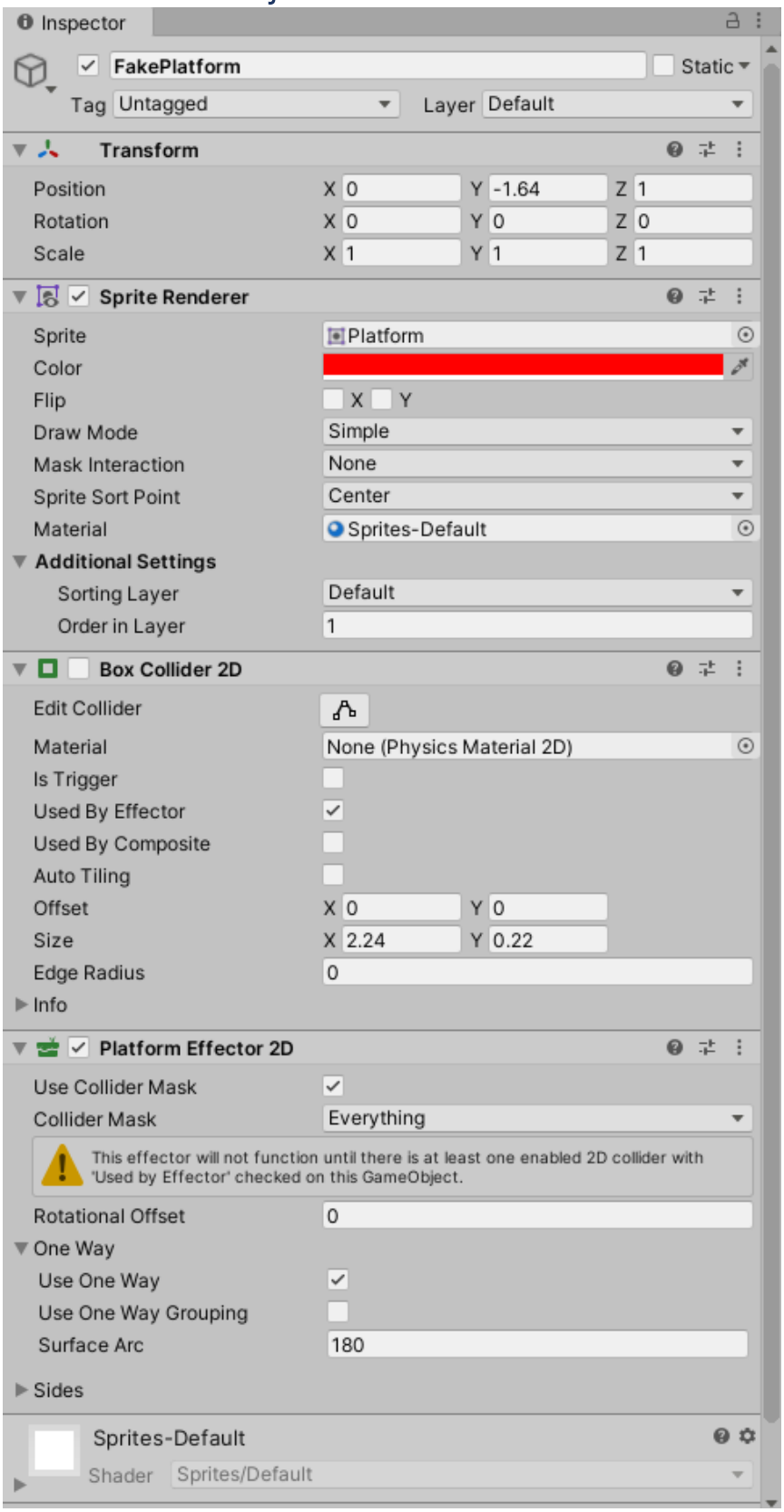
Hierarchy



FakeGameController Object

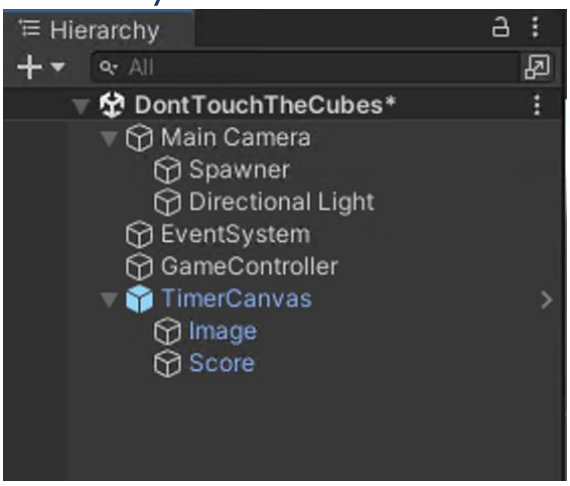


FakePlatform Object



Activity Solution: Don't Touch the Cubes

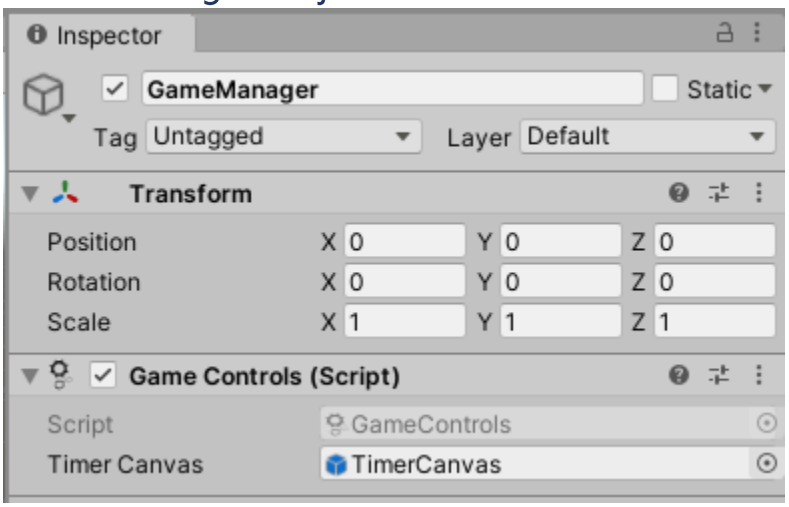
Hierarchy



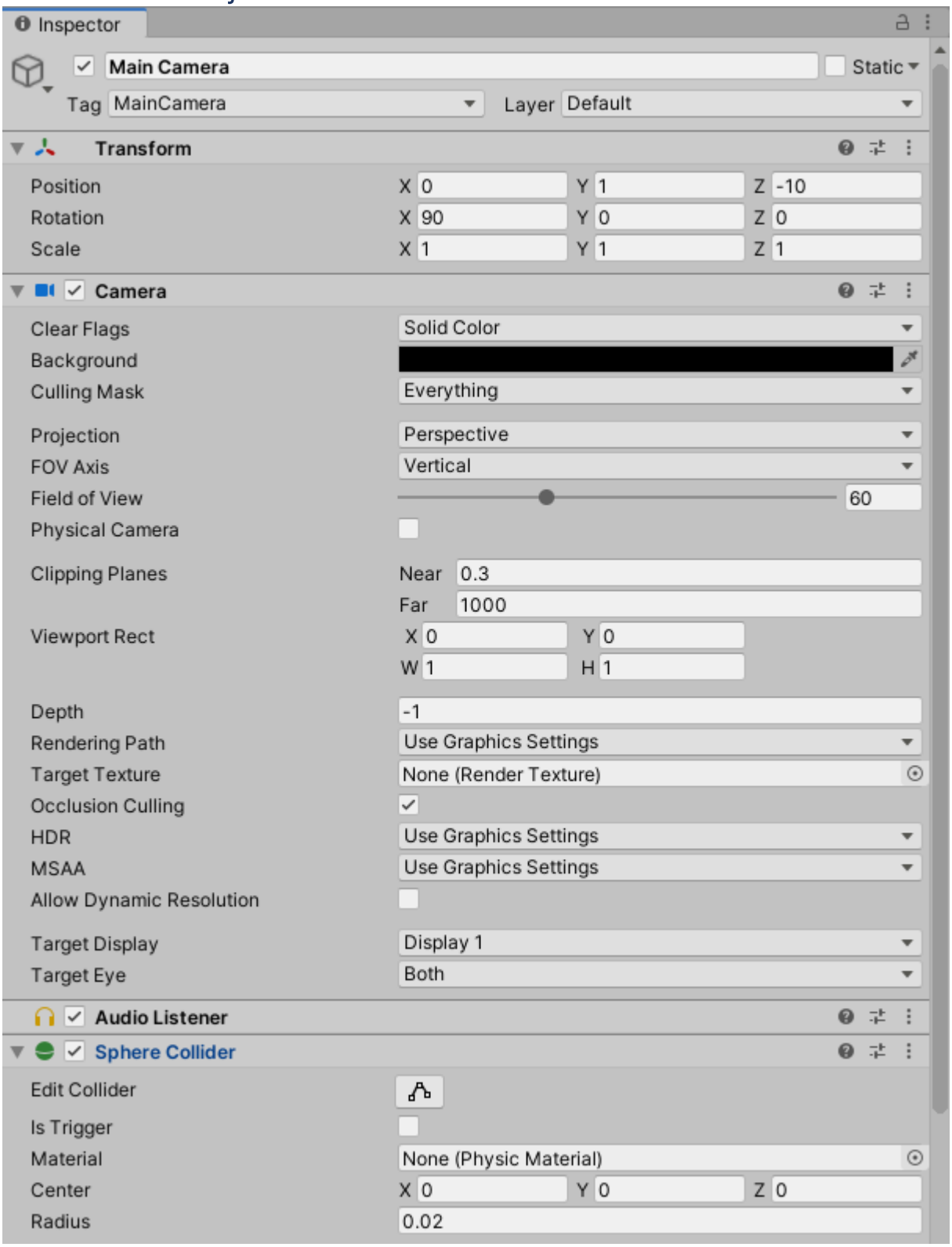
GameControls.cs Script

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

public class GameControls : MonoBehaviour
{
    public GameObject timerCanvas;
    private Text timerText;
    private int timerCount;

    void Start()
    {
        timerCanvas.SetActive(true);
        StartCoroutine(CountTime());
        Time.timeScale = 1f;
        timerText = GameObject.Find("Score").GetComponent<Text>();
    }

    IEnumerator CountTime()
    {
        // add one point every second
        yield return new WaitForSeconds(1f);
        timerCount++;
        timerText.text = "Score: " + timerCount;
        StartCoroutine(CountTime());
    }
}
```

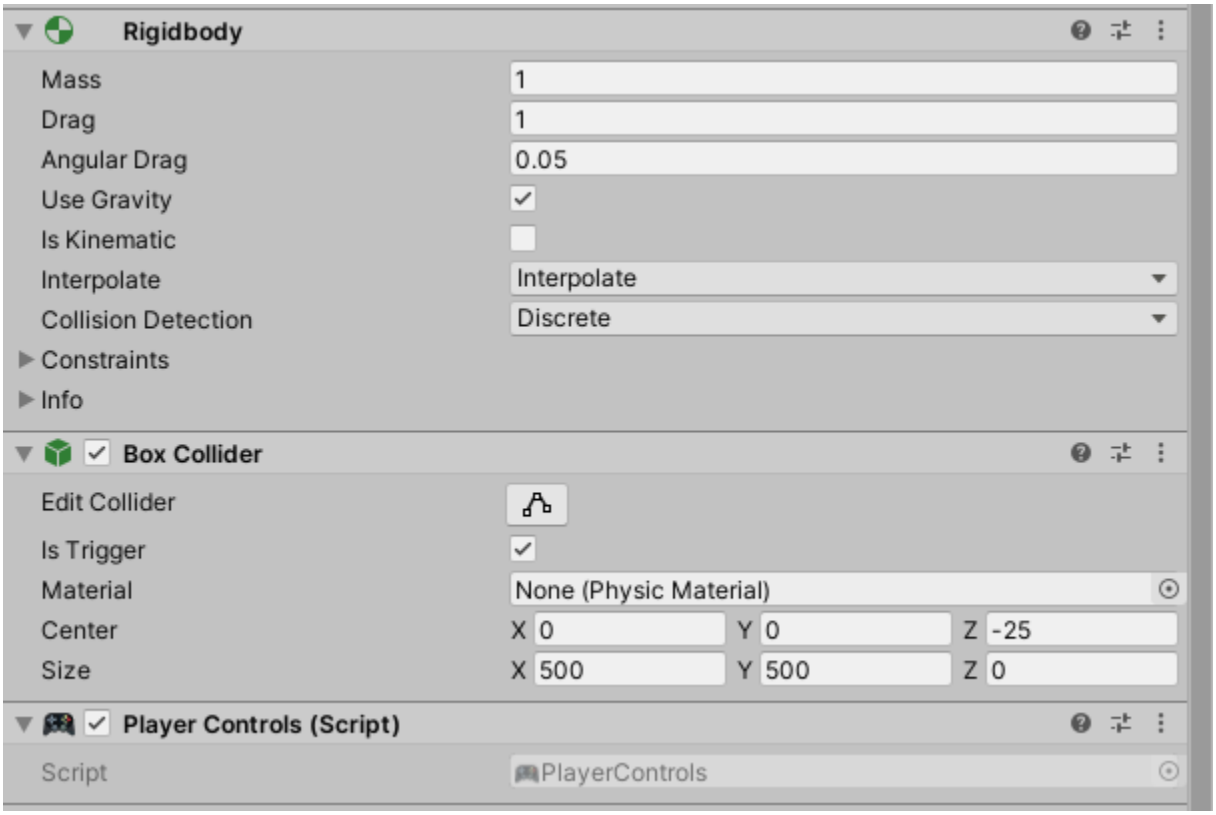
GameManager Object



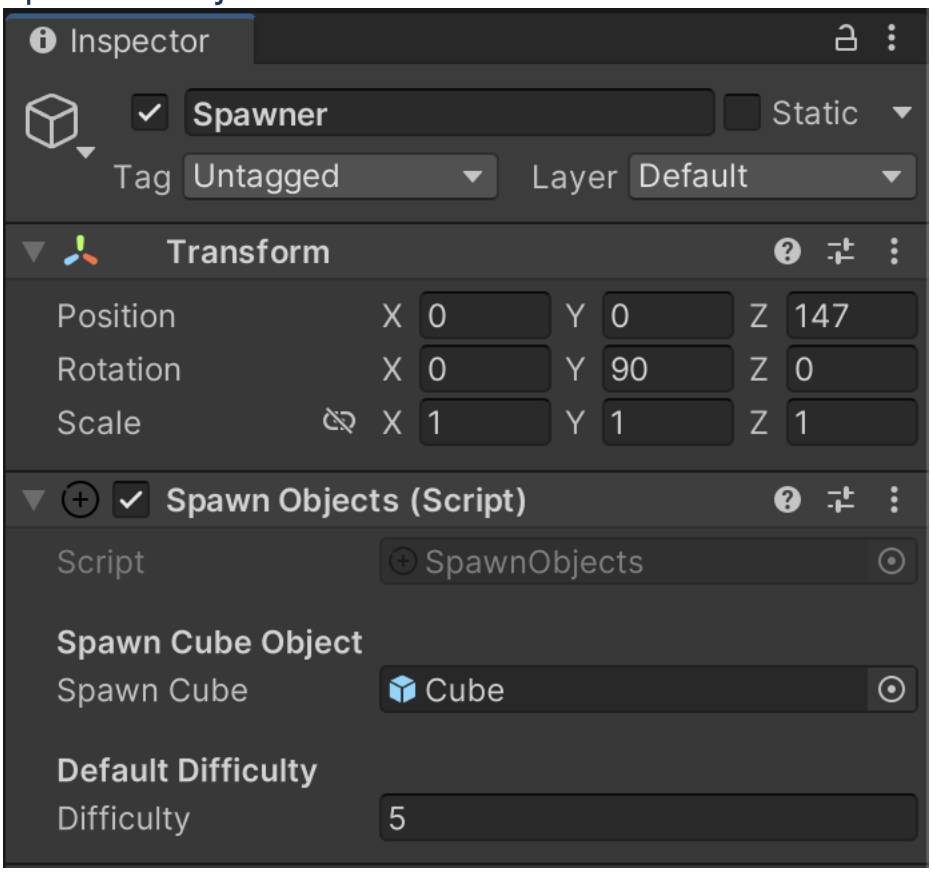
Main Camera Object



Main Camera Object Continued:



Spawner Object

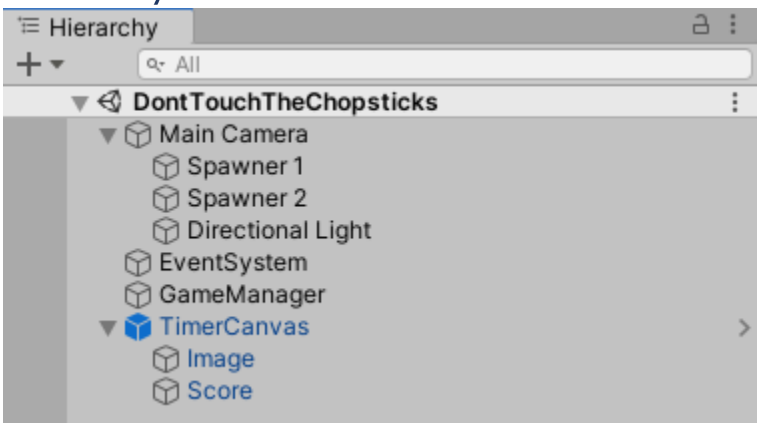


SpawnObjects.cs Script

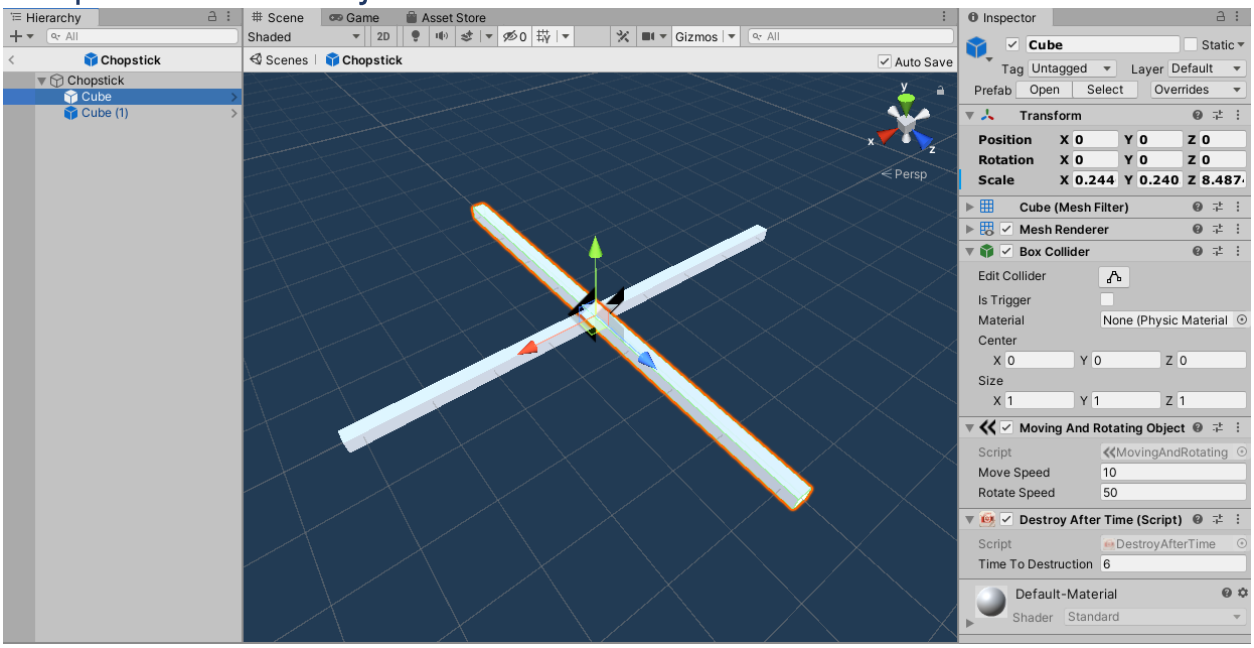
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class SpawnObjects : MonoBehaviour
{
    public GameObject spawnCube;
    public float difficulty = 40f;
    // time delay between spawns
    float spawn;

    void Update()
    {
       spawn += difficulty * Time.deltaTime;
       while(spawn > 0)
       {
              spawn -= 1:
              Vector v3Pos = transform.position + new Vector (Random. value * 40f - 20f,
       0, Random.value * 40f - 20f);
       Quaternion Rotation = Quaternion.Euler(0, Random.value * 360f, Random.value *
30f);
       GameObject createObject = Instantiate(spawnCube, V3Pos, qRotation);
    }
}
```

Activity Solution: Don't Touch the Chopsticks Prove Yourself

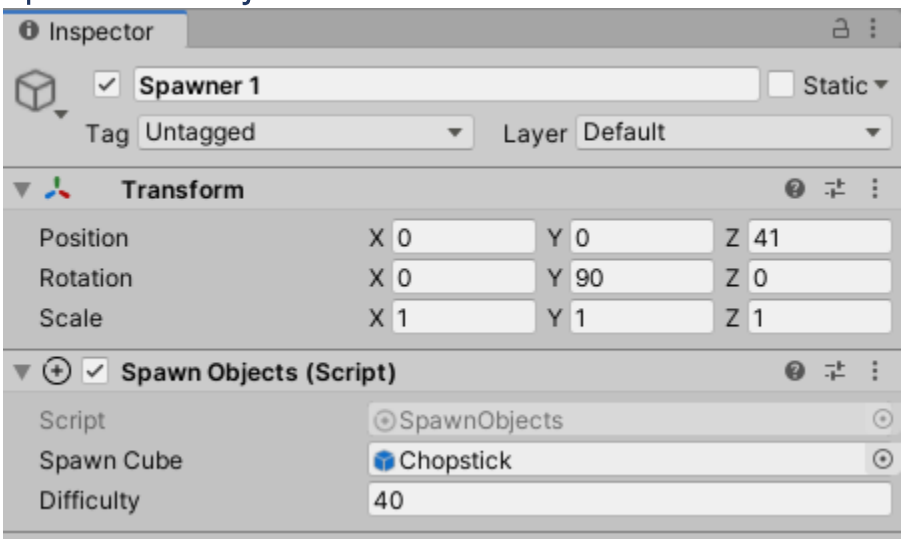
Hierarchy



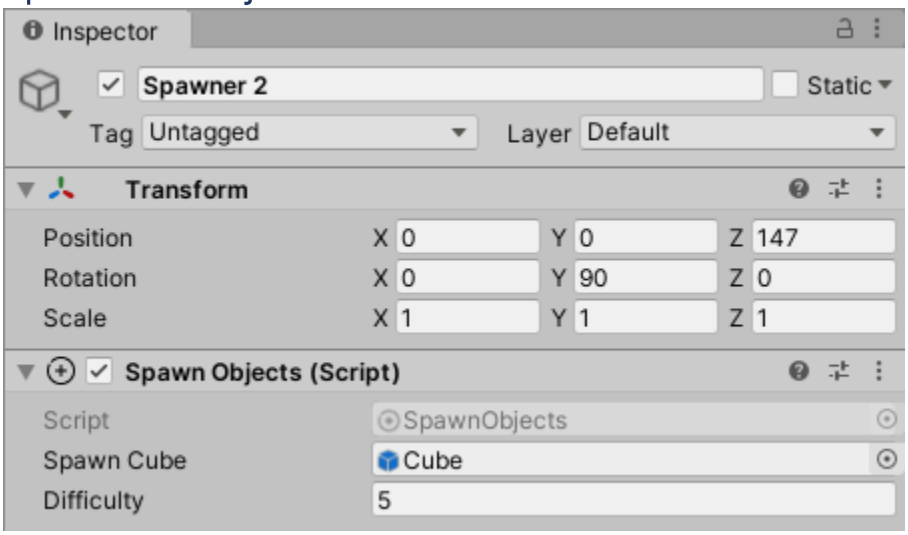
Chopstick Prefab Object



Spawner 1 Object

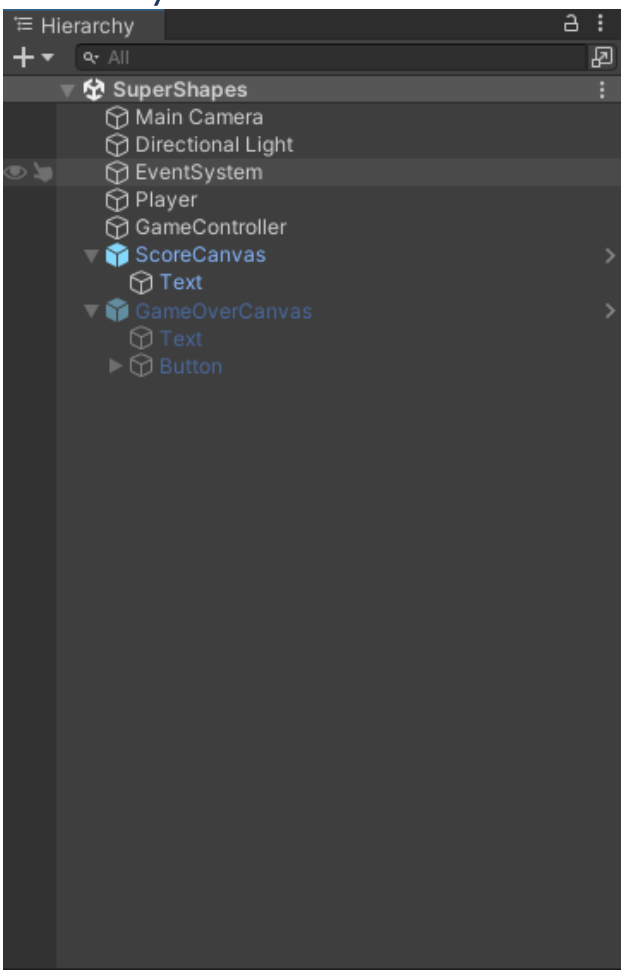


Spawner 2 Object

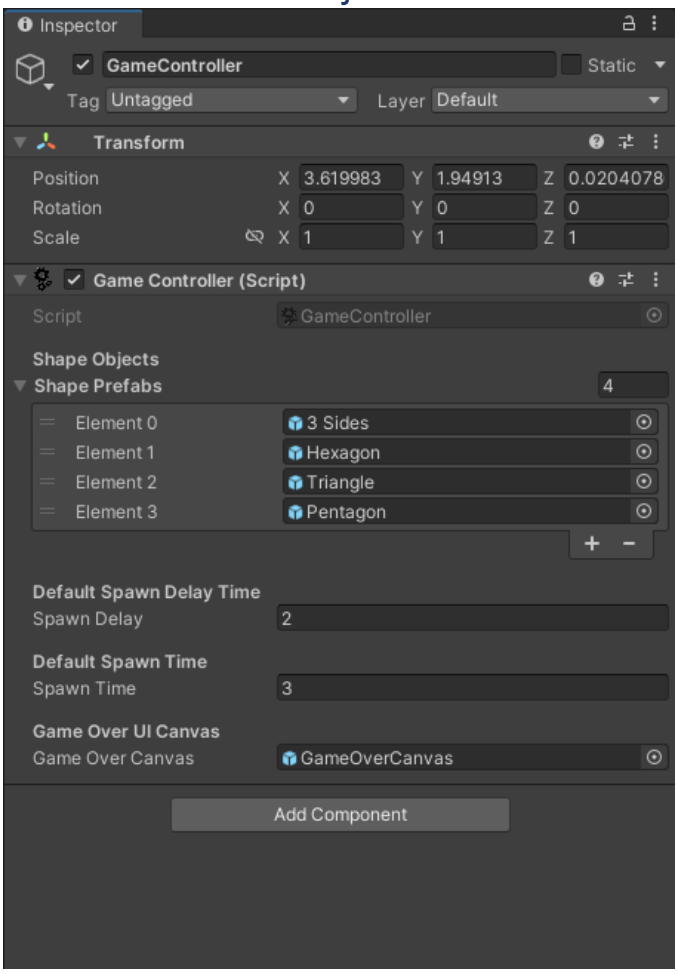


Activity Solution: Super Shapes

Hierarchy



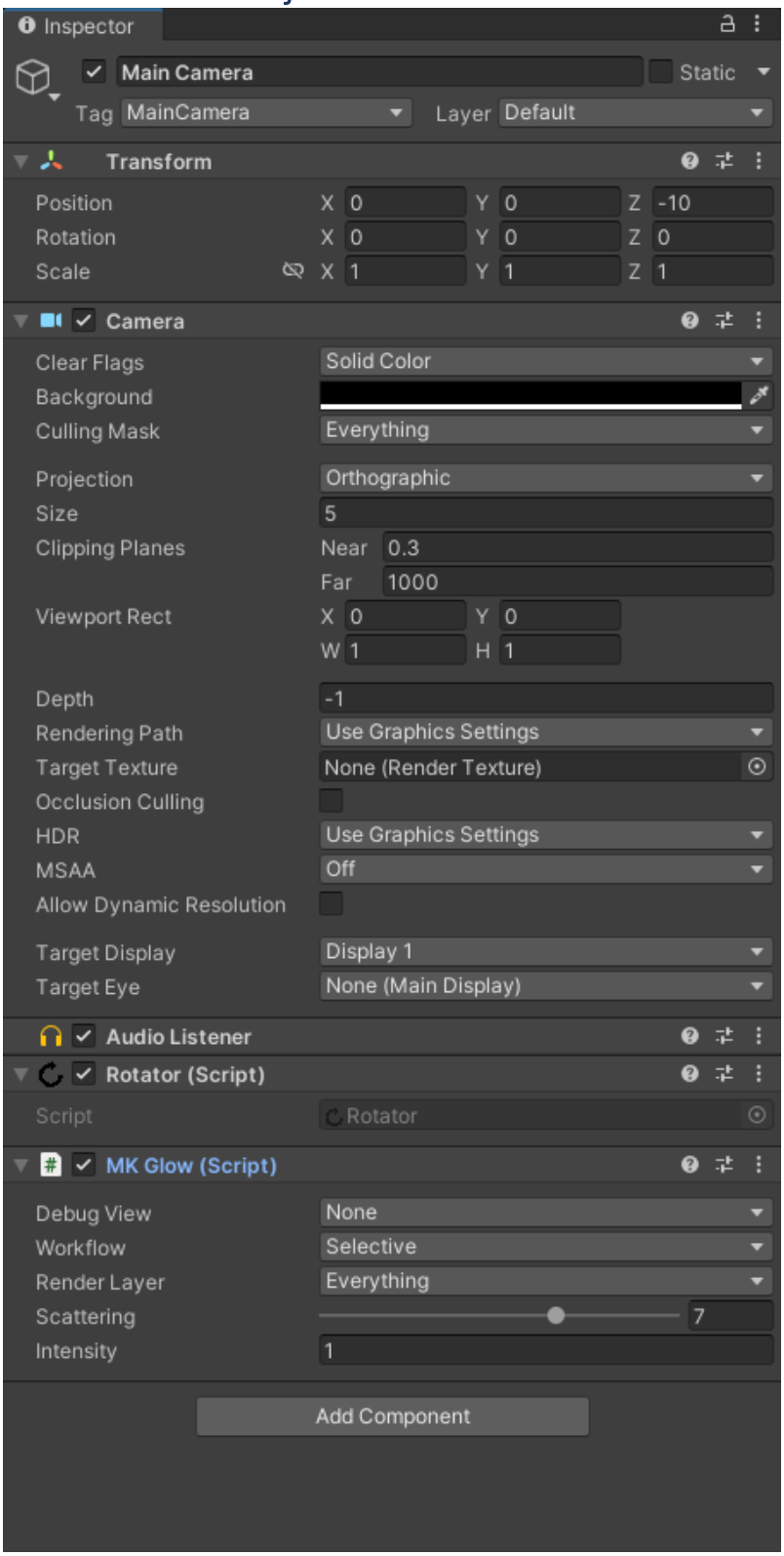
GameController Object



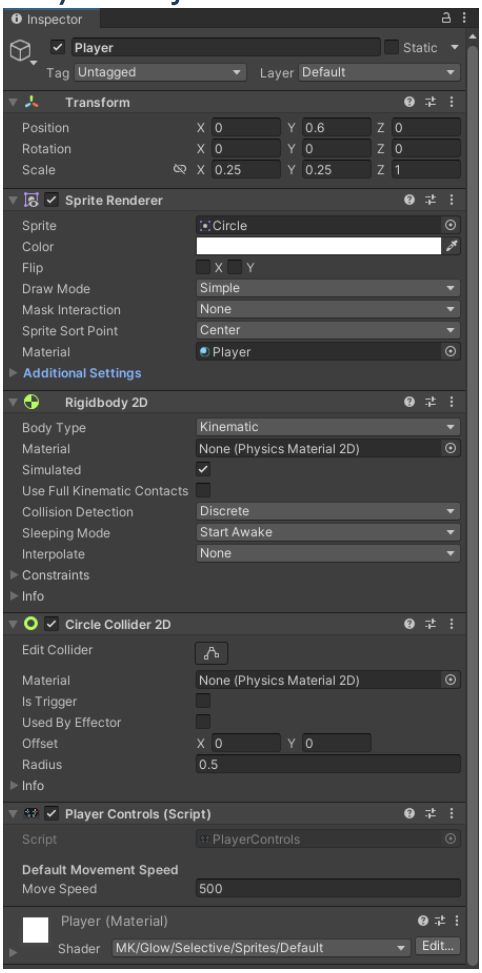
GameController.cs Script

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
public class GameController : MonoBehaviour
{
	//The [] tells Unity that we want an Array
	[Header("Shape Objects")]
	public GameObject[] shapePrefabs;
	//The first object will spawn after
	//the spawnDelay and then every spawnTime
	[Header("Default Spawn Delay Time")]
	public float spawnDelay = 2;
	[Header("Default Spawn Time")]
	public float spawnTime = 3;
	[Header("Game Over UI Canvas")]
	public GameObject gameOverCanvas;
	// Start is called before the first frame update
	void Start()
	{
		InvokeRepeating("Spawn", spawnDelay, spawnTime);
	}
	void Spawn()
	{
		//Get a random number between the range of 0 and our
		//array length
		int randomInt = Random.Range(0, shapePrefabs.Length);
		//spawn new hexagon that was picked randomly
		Instantiate(shapePrefabs[randomInt],Vector3.zero,Quaternion.identity);
	}

	public void GameOver()
	{
		CancelInvoke("Spawn");
		//Set the gameOverCanvas to be visible
		gameOverCanvas.SetActive(true);
		Time.timeScale = 0;
	}
}
```

Main Camera Object



Player Object

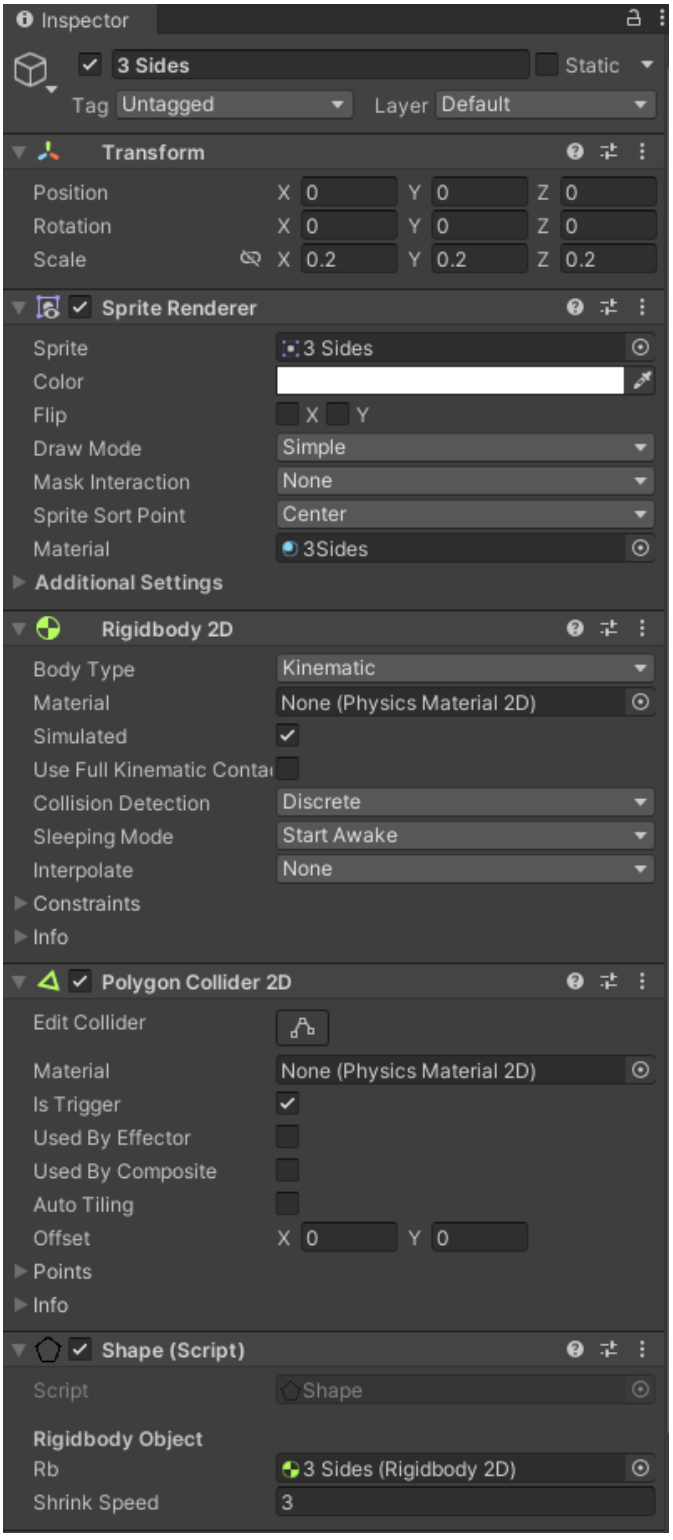


PlayerControls.cs Script

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


public class PlayerControls : MonoBehaviour
{
	//movement speed
	[Header("Default Movement Speed")]
	public float moveSpeed = 200;
	//movement float
	float movement = 0;
	// Start is called before the first frame update
	void Start()
	{
		//Timescale is the speed the game runs at
		Time.timeScale = 1;
	}
	// Update is called once per frame
	void Update()
	{
		//Movement equal to Left and Right input
		movement = Input.GetAxisRaw("Horizontal");
	}
	private void FixedUpdate()
	{
		//Object transformation rotates around
		//an object using the settings.
		transform.RotateAround(Vector3.zero, Vector3.forward, movement *
	Time.fixedDeltaTime * -moveSpeed);
	}


	//This function runs if we overlap with a collider set to Trigger
	private void OnTriggerEnter2D(Collider2D collision)
	{
		GameObject.Find("GameController").GetComponent<GameController>().GameOver()
	;
	}
}
```

Shape Objects

Each shape will have a different name and Rb property in the Shape script component.



Shape.cs Script

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class Shape : MonoBehaviour
{
	//Rigidbody for the object
	[Header("Rigidbody Object")]
	public Rigidbody2D rb;
	//Shrinking Speed
	public float shrinkSpeed = 3;
	// Start is called before the first frame update
	void Start()
	{
		//Rotation of the rigidbody
		//at a random range
		rb.rotation = Random.Range(0, 360);
		//local scale for the Hexagon
		//equals one for all axes (x,y,z) times ten
		//meaning - localScale = (1,1,1) * 10
		transform.localScale = Vector3.one * 10;
	}


	// Update is called once per frame
	void Update()
	{
		//slowly shrink our localScale by
		//our shrinkSpeed multiplied by game rate
		transform.localScale -= Vector3.one * shrinkSpeed * Time.deltaTime;

		//When the object gets too small
		if (transform.localScale.x < 0.05f)
		{
			//Destroy Object
			Destroy(gameObject);
			Score.score++;
		}
	}
}
```

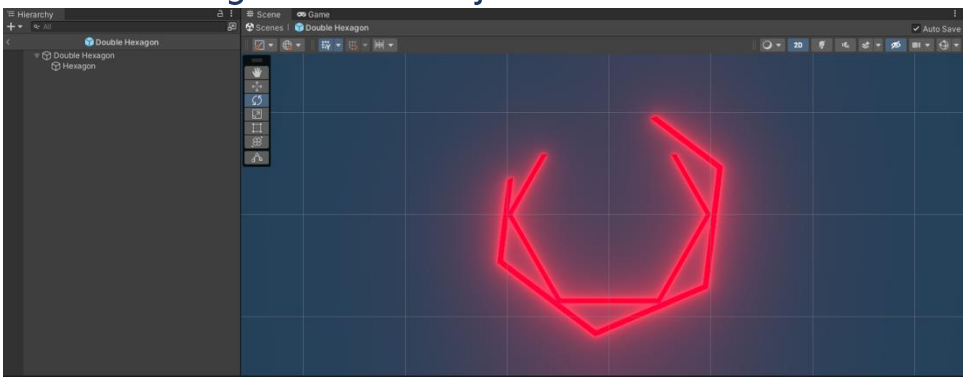
Rotator.cs Script

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

public class Rotator : MonoBehaviour
{
	// Update is called once per frame
	void Update()
	{
		transform.Rotate(Vector3.forward, Time.deltaTime * 30);
	}
}
```

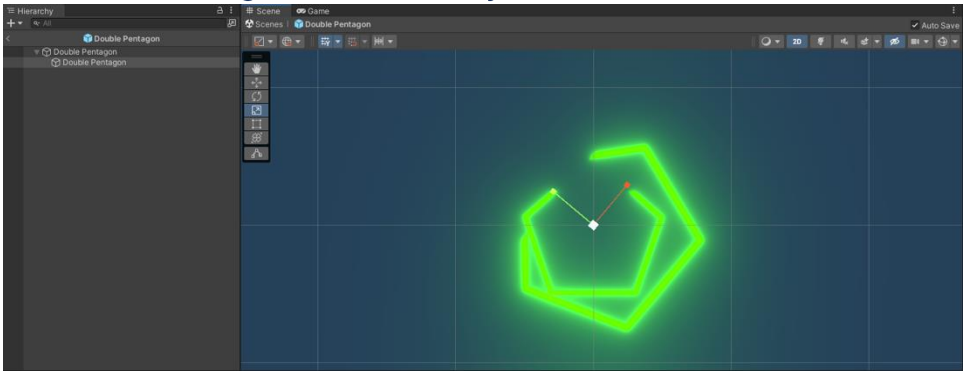
Activity Solution: Super Duper Shapes Prove Yourself

Each Shape may be different. Ninjas may have combined multiple shapes or created new and interesting challenges. Below are examples of what could be created.

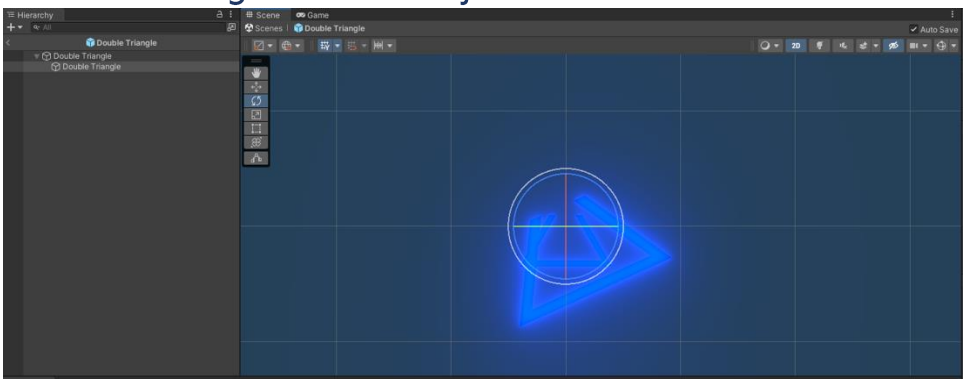
Double Hexagon Prefab Object



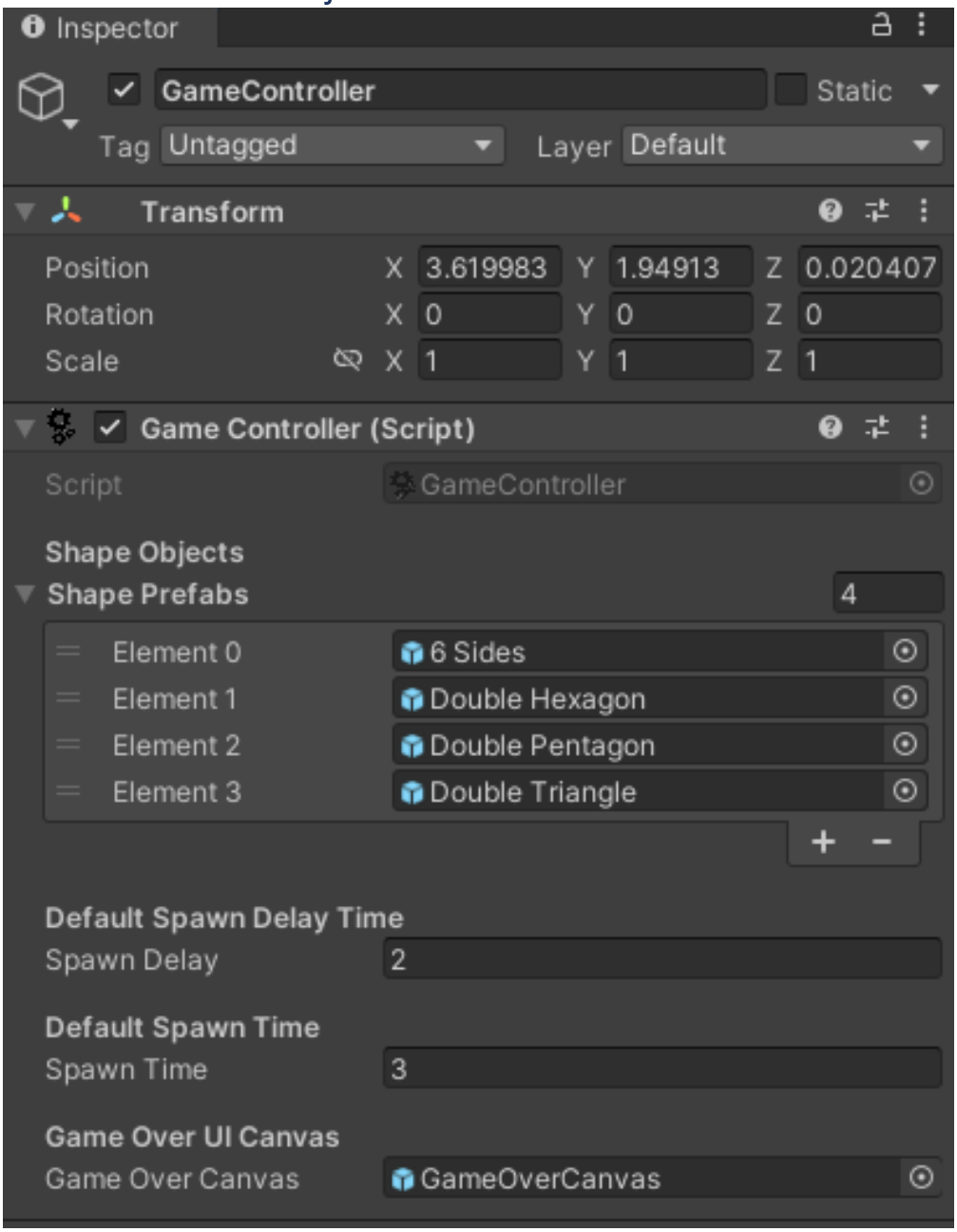
Double Pentagon Prefab Object



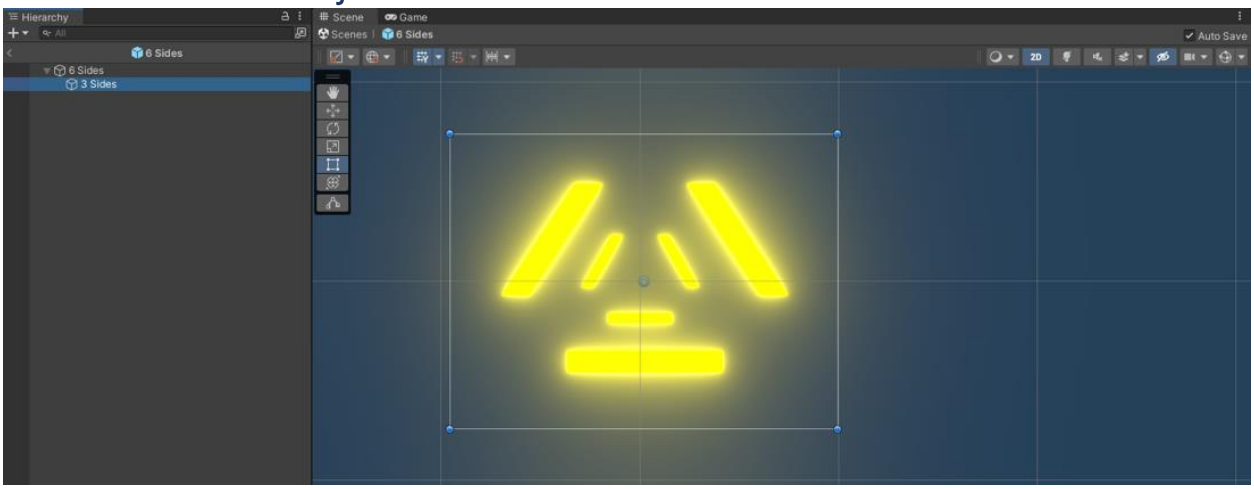
Double Triangle Prefab Object



GameController Object

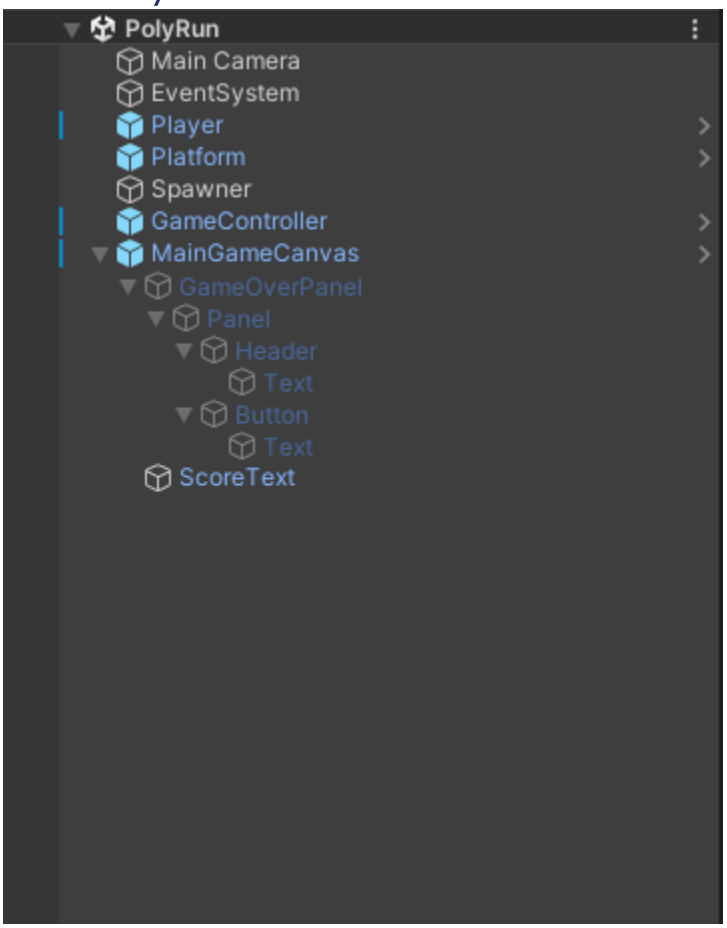


Six Sides Prefab Object

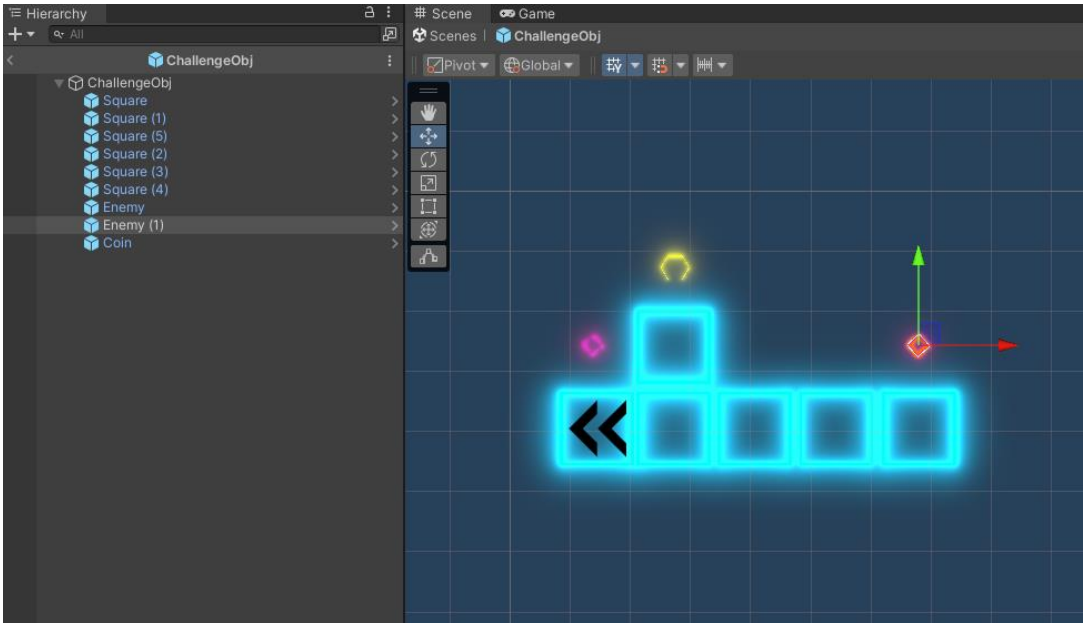


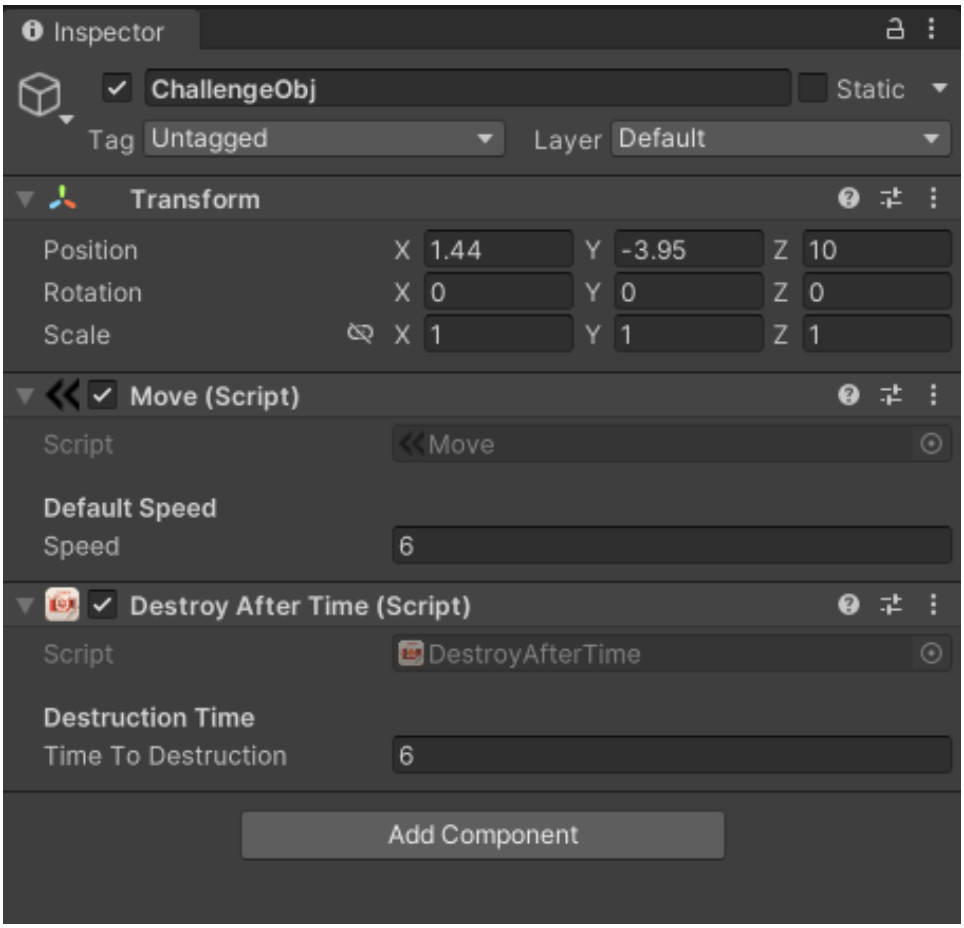
Activity Solution: Polyrun

Hierarchy



ChallengeObj Prefab Object

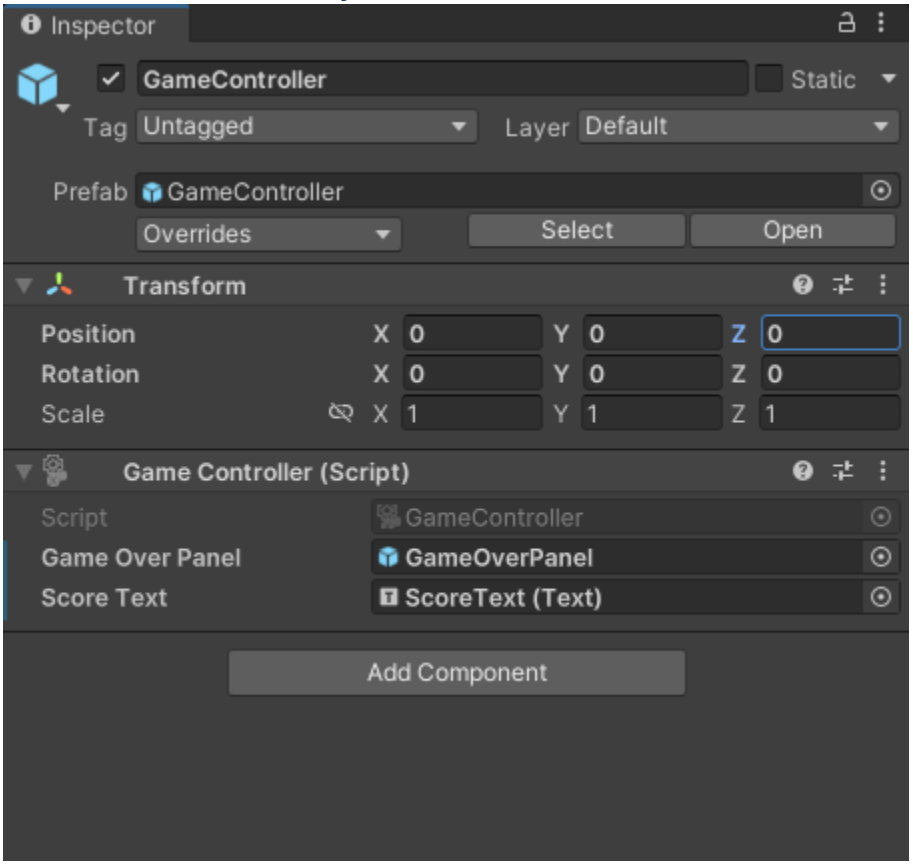




DestroyAfterTime.cs Script

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class DestroyAfterTime : MonoBehaviour
{
	[Header("Destruction Time")]
	public float timeToDestruction = 6;
	// Start is called before the first frame update
	void Start()
	{
		Invoke("DestroyObject", timeToDestruction);
	}
	void DestroyObject()
	{
		Destroy(gameObject);
	}
}
```

GameController Object



GameController.cs Script

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

public class GameController : MonoBehaviour
{
	public GameObject gameOverPanel;
	public Text scoreText;
	int score = 0;


	public void GameOver()
	{
		Time.timeScale = 0;
		scoreText.gameObject.SetActive(false);
		gameOverPanel.SetActive(true);
	}


	public void IncrementScore()
	{
		score++;
		scoreText.text = score.ToString();
	}
}
```

Move.cs Script

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

public class Move : MonoBehaviour
{
	[Header("Default Speed")]
	public float speed = 6;
	// Update is called once per frame
	void Update()
	{
		//Add vector3.left (-1,0,0) to our transform, times by speed
		transform.position += Vector3.left * speed * Time.deltaTime;
	}
}
```

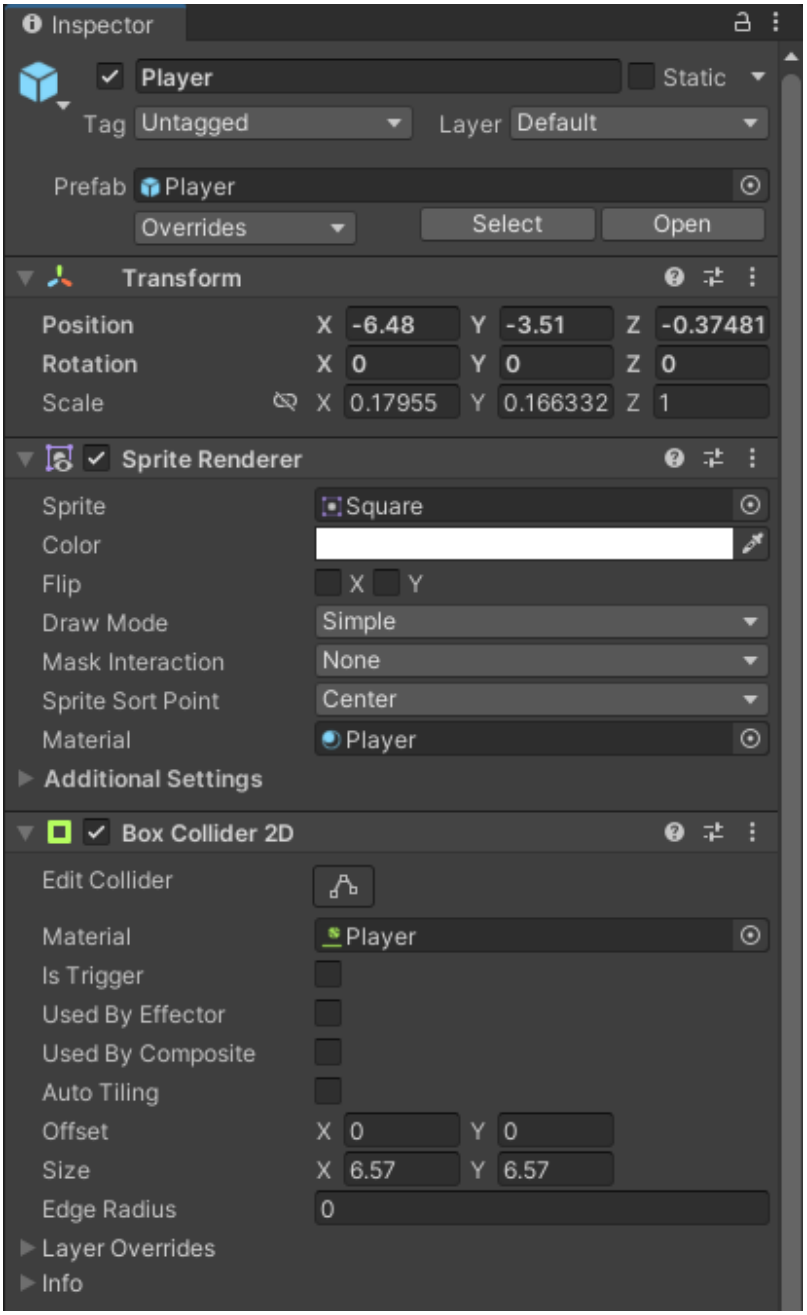
PlayerControls.cs Script

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class PlayerControls : MonoBehaviour
{
	//Jumping power for the player object
	[Header("Default Jumping Power")]
	public float jumpPower = 6f;
	//True or false if is on the ground
	[Header("Boolean isGrounded")]
	public bool isGrounded;
	//position of the object
	float posX = 0.0f;
	//rigibody2D of the object
	Rigidbody2D rb;


	// Start is called before the first frame update
	void Start()
	{
		//Variable rb equals to Rigidbody2D
		//component on the object
		//this script is attached to
		rb = GetComponent<Rigidbody2D>();
		//posX = starting position
		posX = transform.position.x;
		Time.timeScale = 1;
	}
	//The function to call when we GameOver
	void GameOver()
	{
		GameObject.Find("GameController").GetComponent<GameController>().GameOver()
		;
		//GameOver()
	}
	// Update is called once per frame
	void Update()
	{
		//Only jump if we press space and if isGrounded is true
		if (Input.GetKeyDown(KeyCode.Space) && isGrounded)
		{
			rb.AddForce(Vector3.up * jumpPower * rb.mass * rb.gravityScale *
		20f);
		}
		//If the player position is
		//less than where it started
		if(transform.position.x < posX)
		{
			//Execute Gameover function
			GameOver();
		}
	}
```

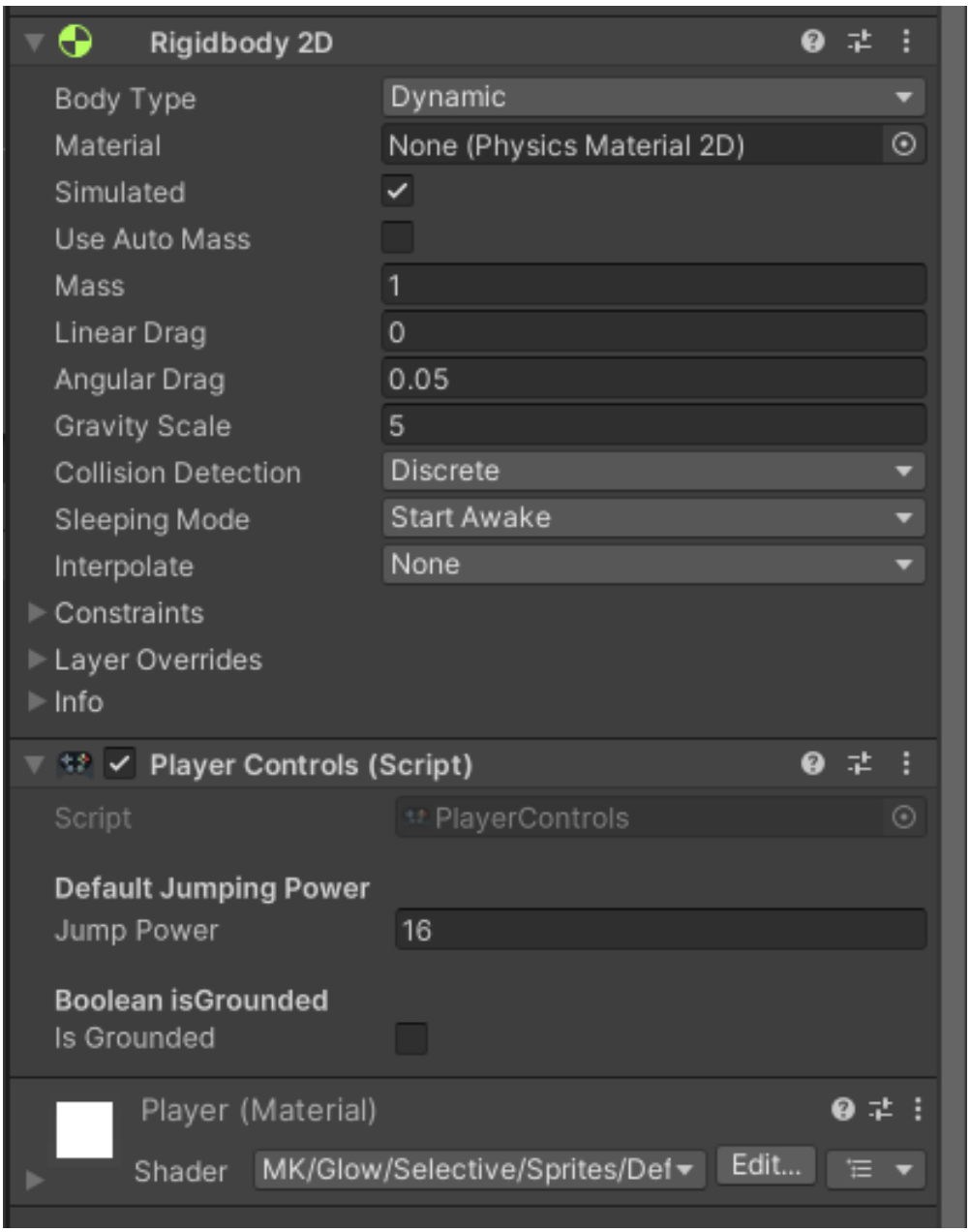
PlayerControls.cs Script Continued

```
        private void OnCollisionEnter2D(Collision2D collision)
        {
                //If the object we collide with has the
                //tag Ground
                if(collision.gameObject.tag == "Ground")
                {
                        //isGrounded is set to true
                        isGrounded = true;
                }
                if(collision.gameObject.tag == "Enemy")
                {
                        GameOver();
                }
        }


        private void OnCollisionStay2D(Collision2D collision)
        {
                //If the object we collide with has the
                //tag Ground
                if (collision.gameObject.tag == "Ground")
                {
                        //isGrounded is set to true
                        isGrounded = true;
                }
        }


        private void OnCollisionExit2D(Collision2D collision)
        {
                //If the object we collide with has the
                //tag Ground
                if (collision.gameObject.tag == "Ground")
                {
                        //isGrounded is set to true
                        isGrounded = false;
                }
        }


        private void OnTriggerEnter2D(Collider2D collision)
        {
                if(collision.tag == "Coin")
                {
                        //Find the game controller and add to our score
                        GameObject.Find("GameController").GetComponent<GameController>().Inc
                rementScore();
                        Destroy(collision.gameObject);
                }
        }


}
```

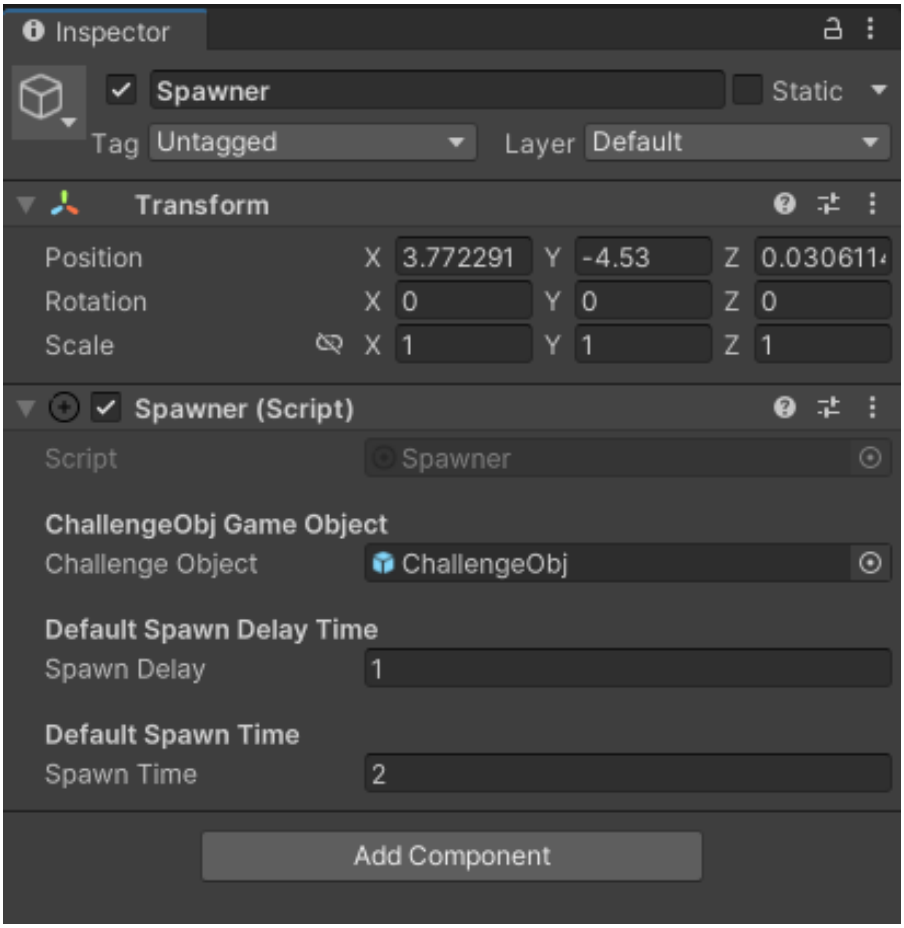
Player Prefab Object



Player Prefab Object Continued



Spawner Object

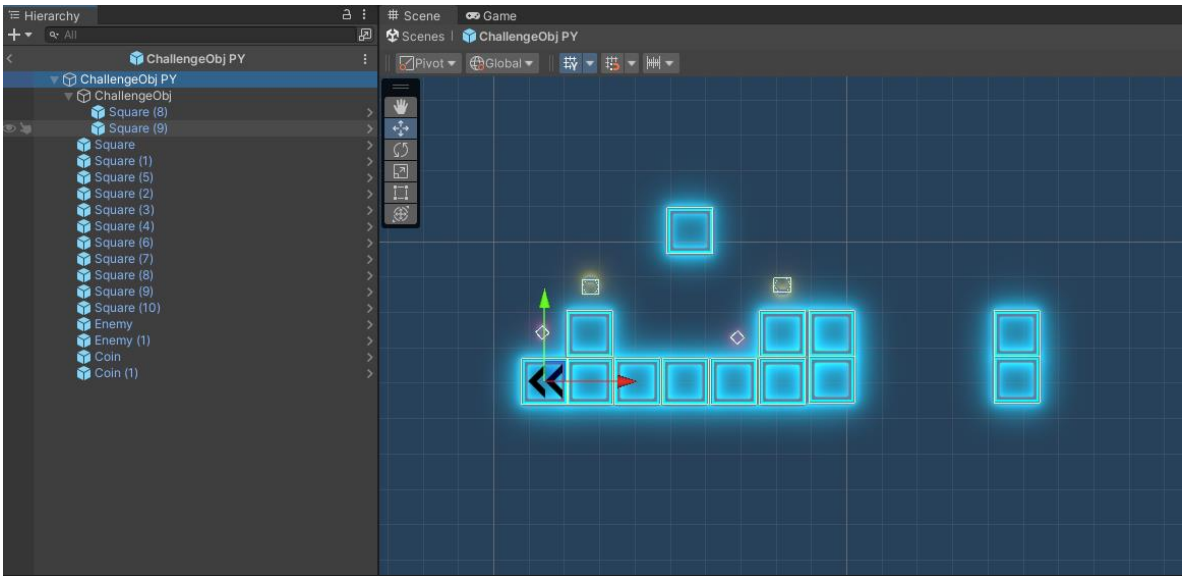


Spawner.cs Script

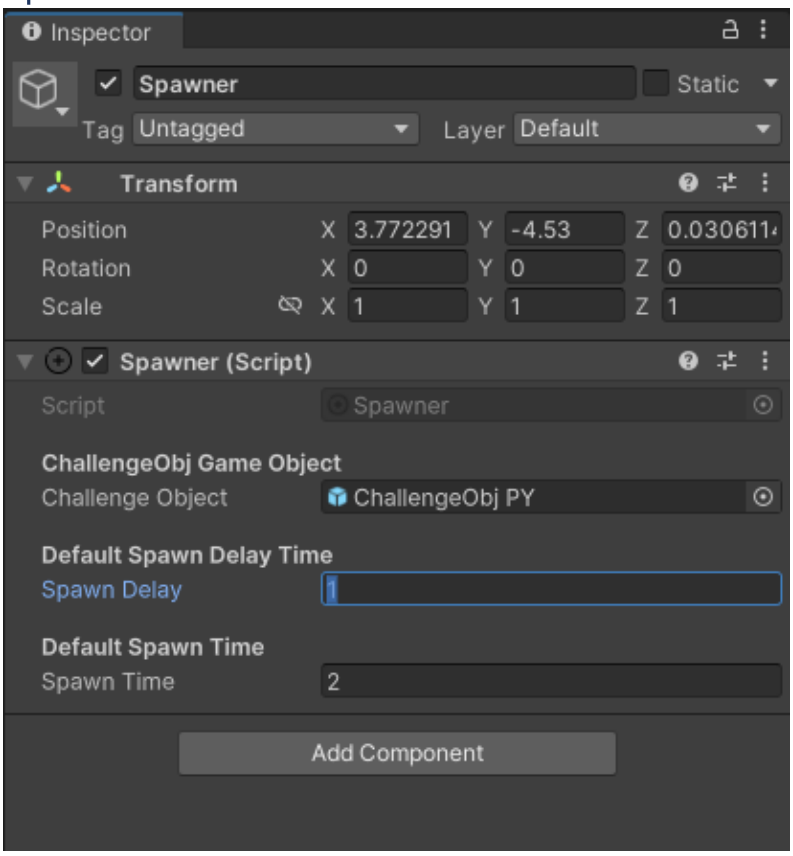
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Spawner : MonoBehaviour
{
	[Header("ChallengeObj Game Object")]
	public GameObject challengeObject;
	[Header("Default Spawn Delay Time")]
	public float spawnDelay = 1f;
	[Header("Default Spawn Time")]
	public float spawnTime = 2f;
	// Start is called before the first frame update
	void Start()
	{
		//start the function after
		//spawnDelay (1) and Repeat the function
		//InstantiateObject every spawnTime (2)
		InvokeRepeating("InstantiateObjects", spawnDelay, spawnTime);
	}


	// Update is called once per frame
	void Update()
	{
		transform.position = new Vector3(15, -3.95f, 0);
	}


	void InstantiateObjects()
	{
		Instantiate(challengeObject, transform.position, transform.rotation);
	}
}
```

Activity Solution: Polyrun v2 Prove Yourself

ChallengeObj Prefab Object

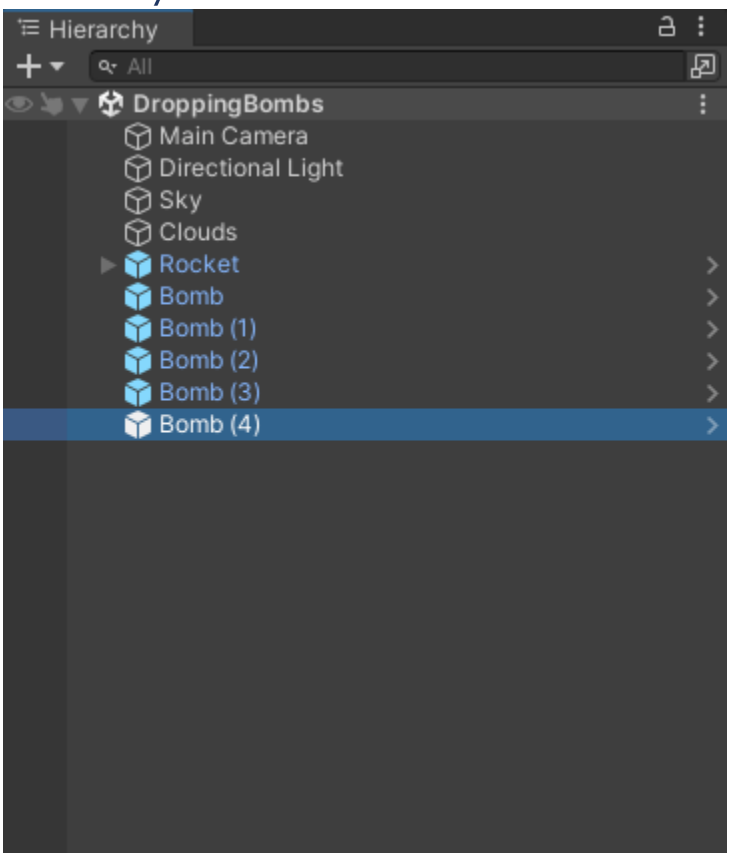


Spawner

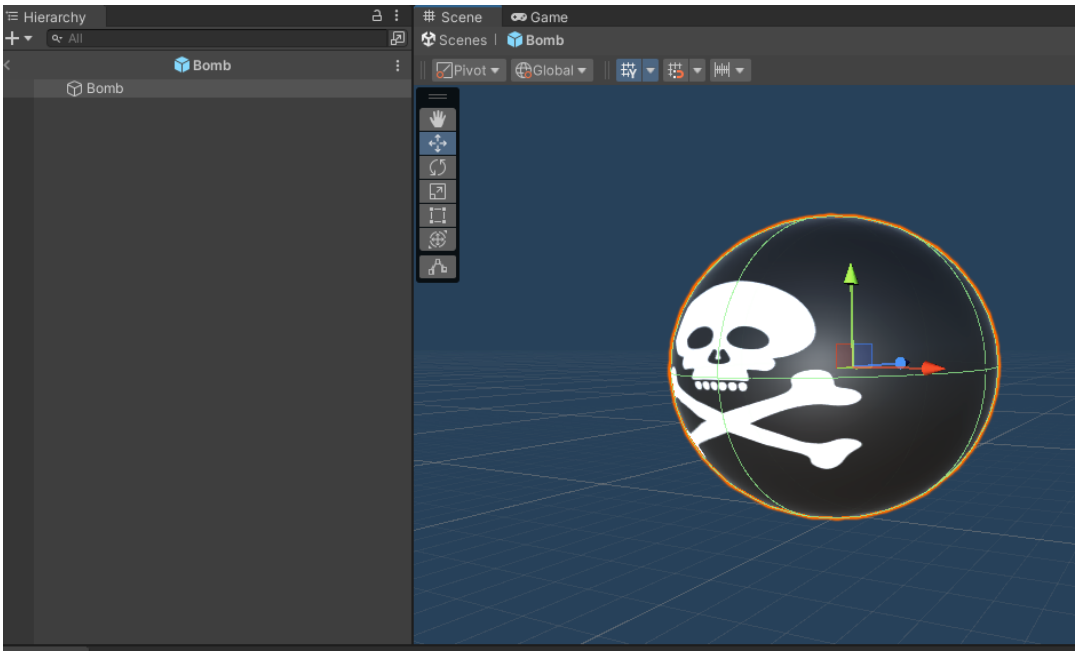


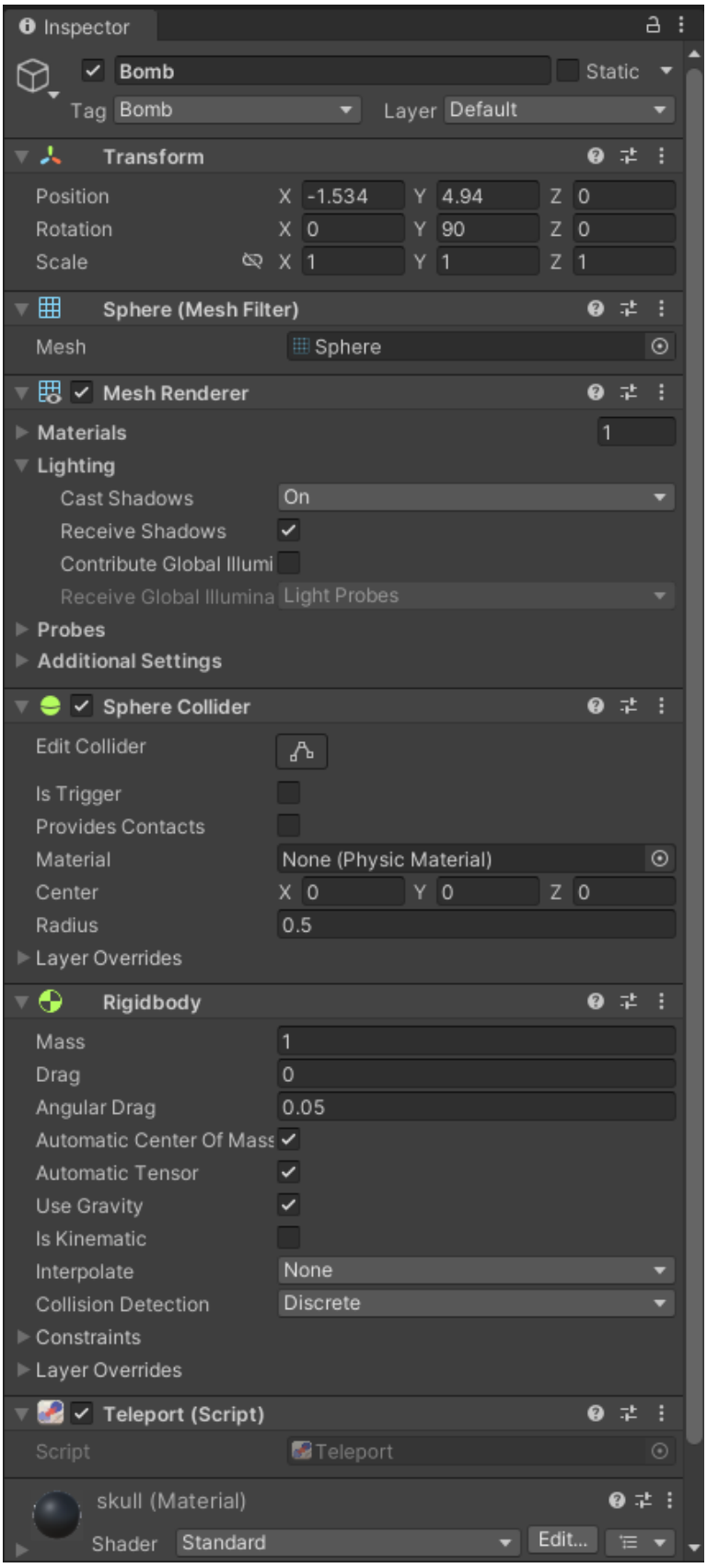
Activity Solution: Dropping Bombs Part 2

Hierarchy

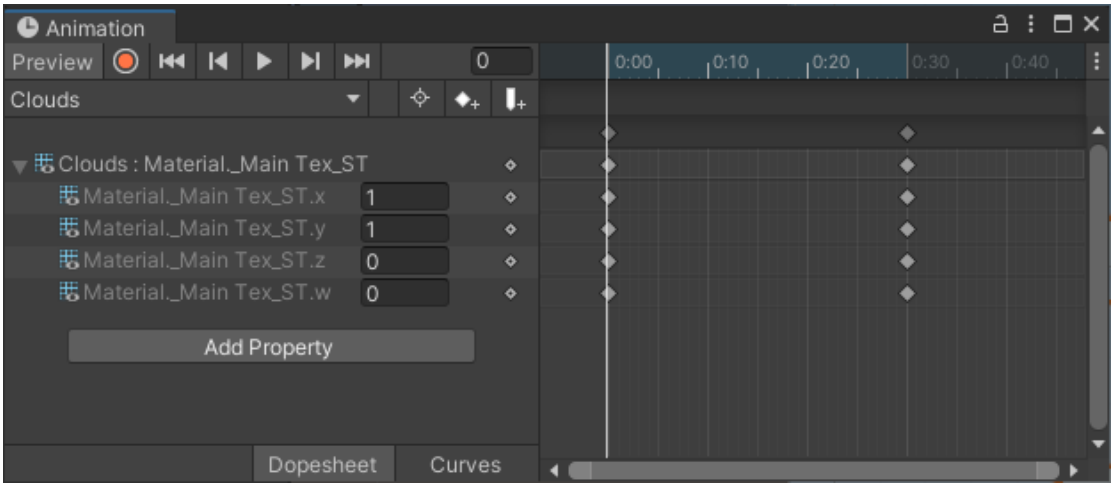


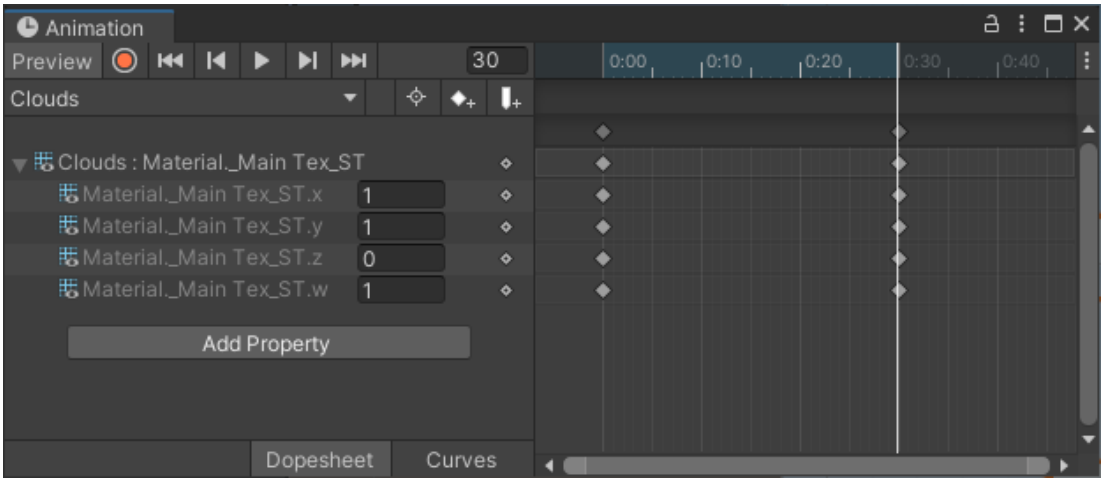
Bomb Prefab Object

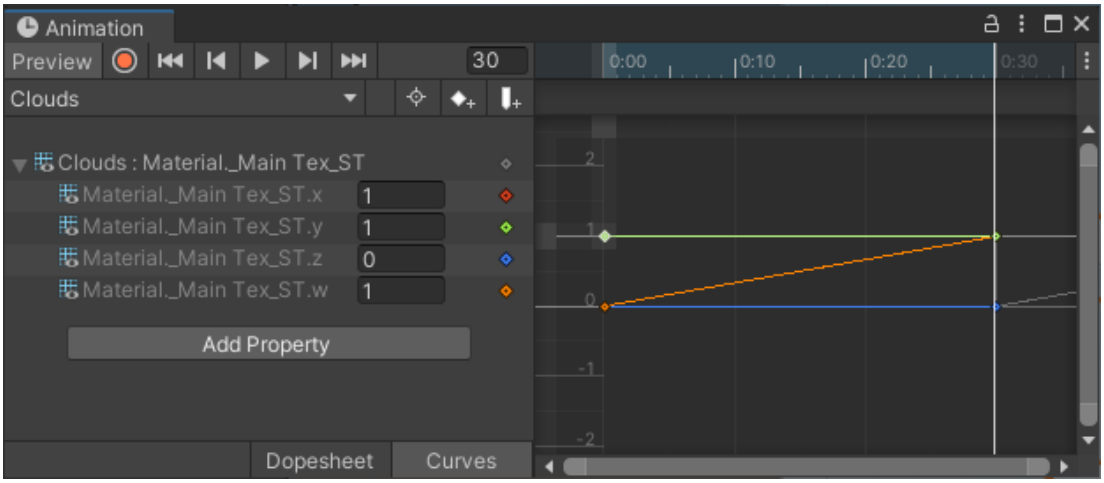




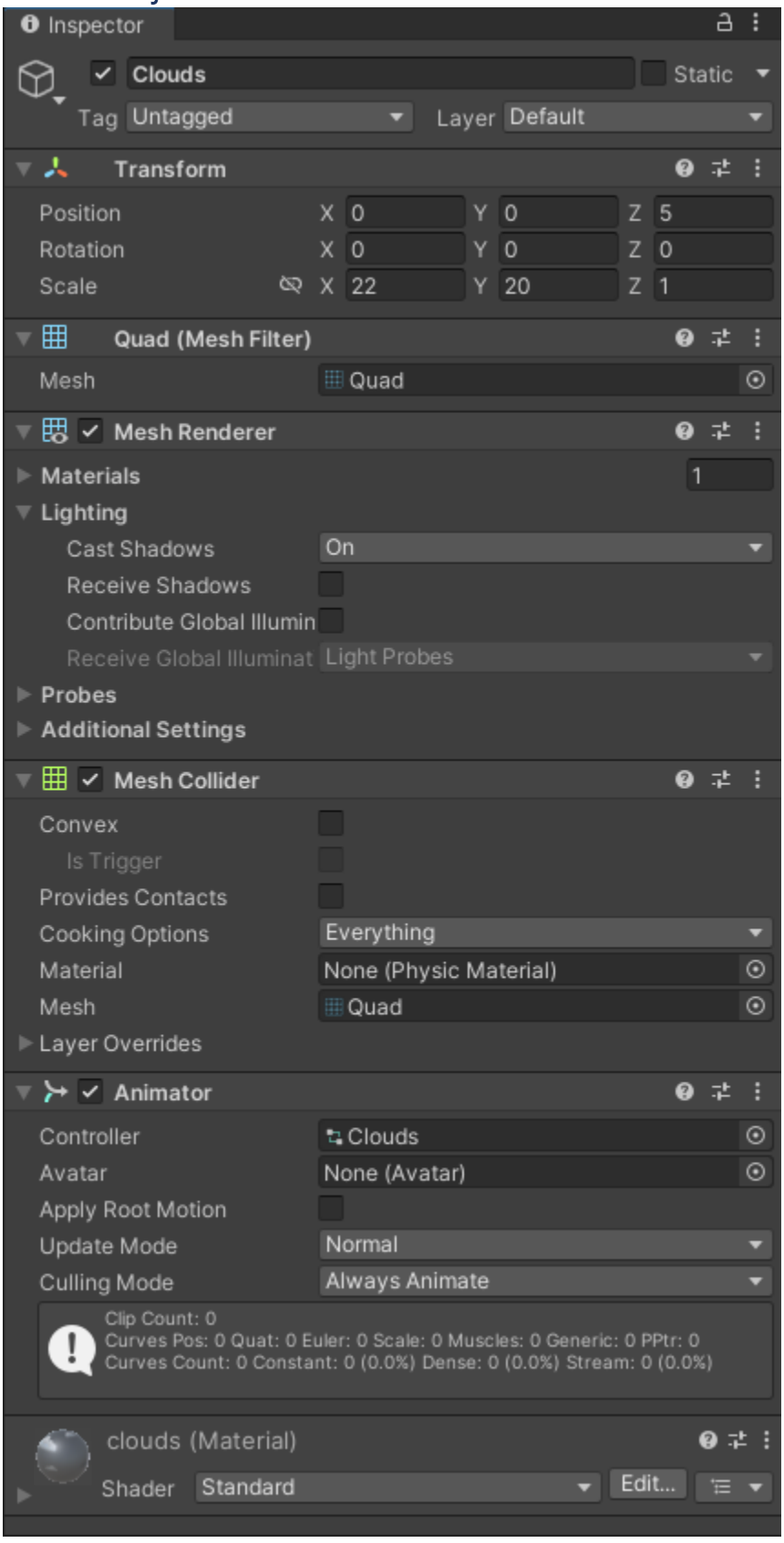
Cloud Animation Frames



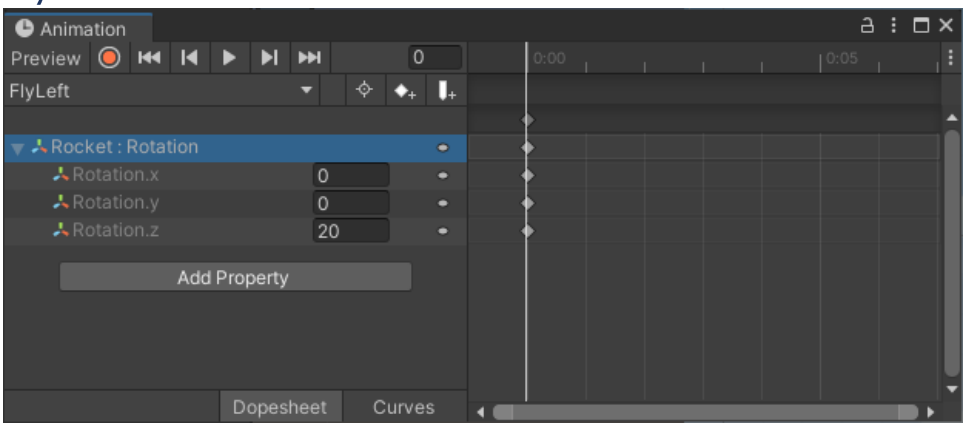




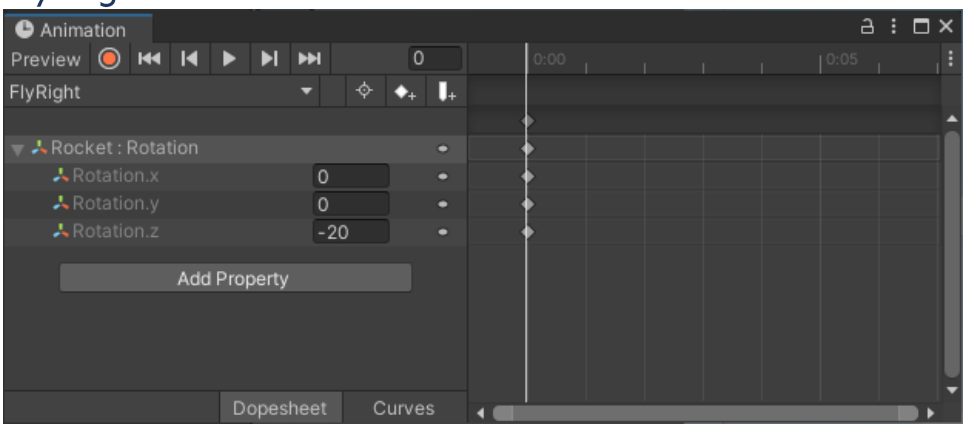
Cloud Object



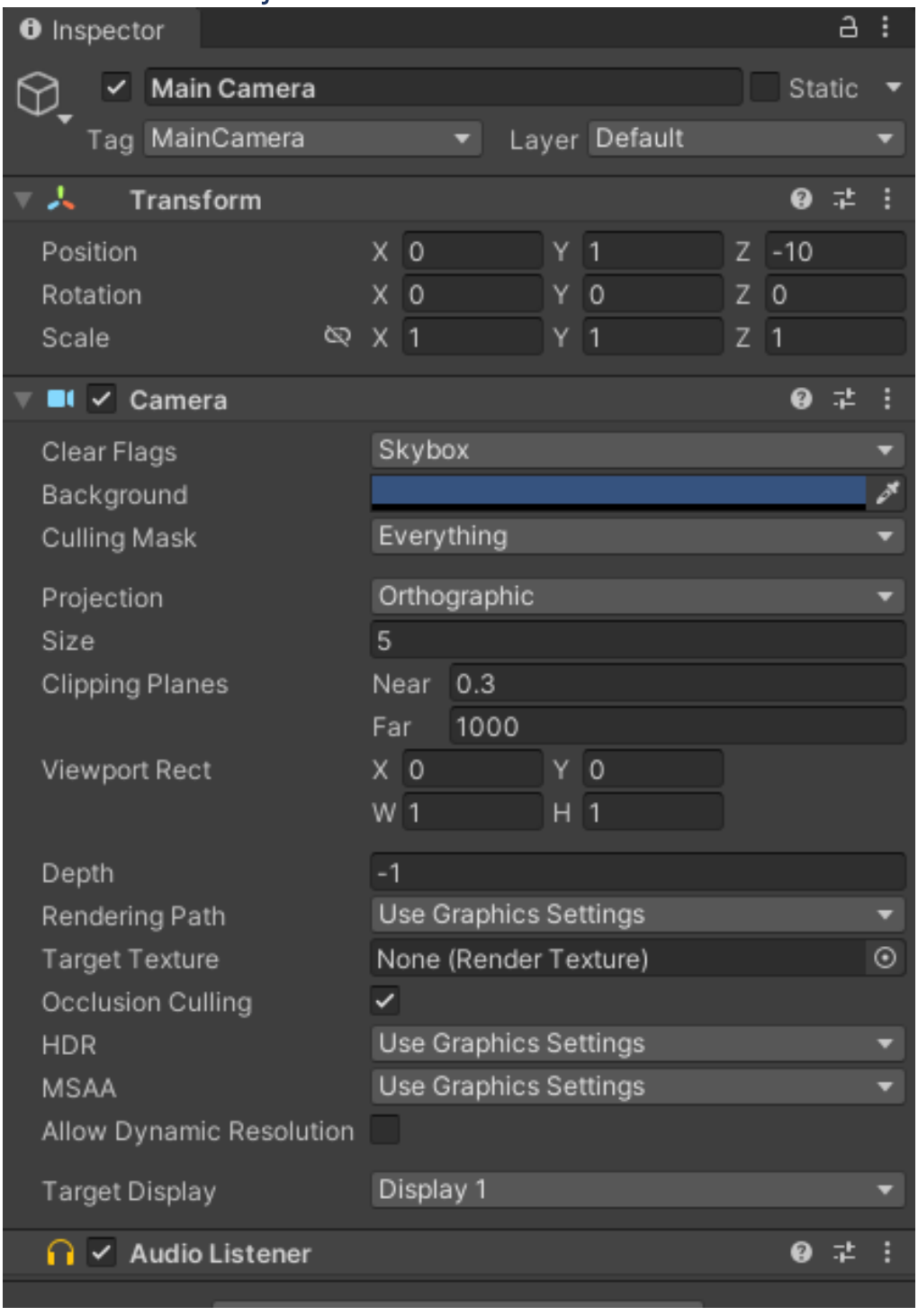
Fly Left Animation Frames



Fly Right Animation Frames



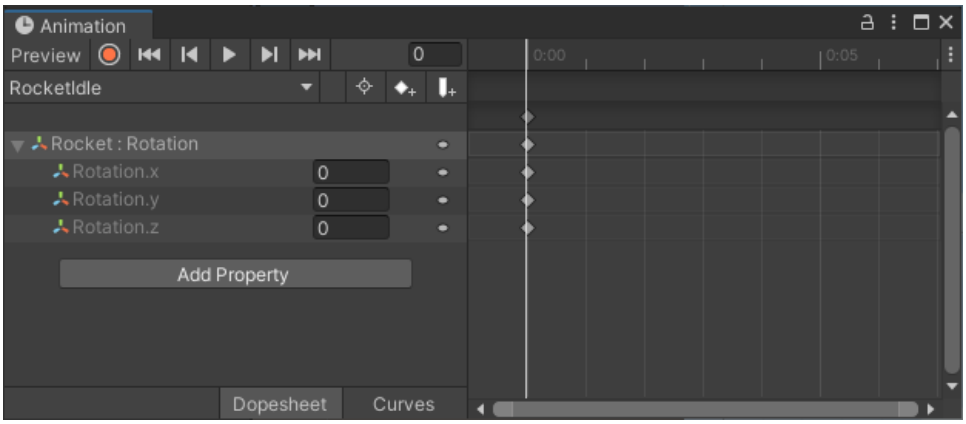
Main Camera Object



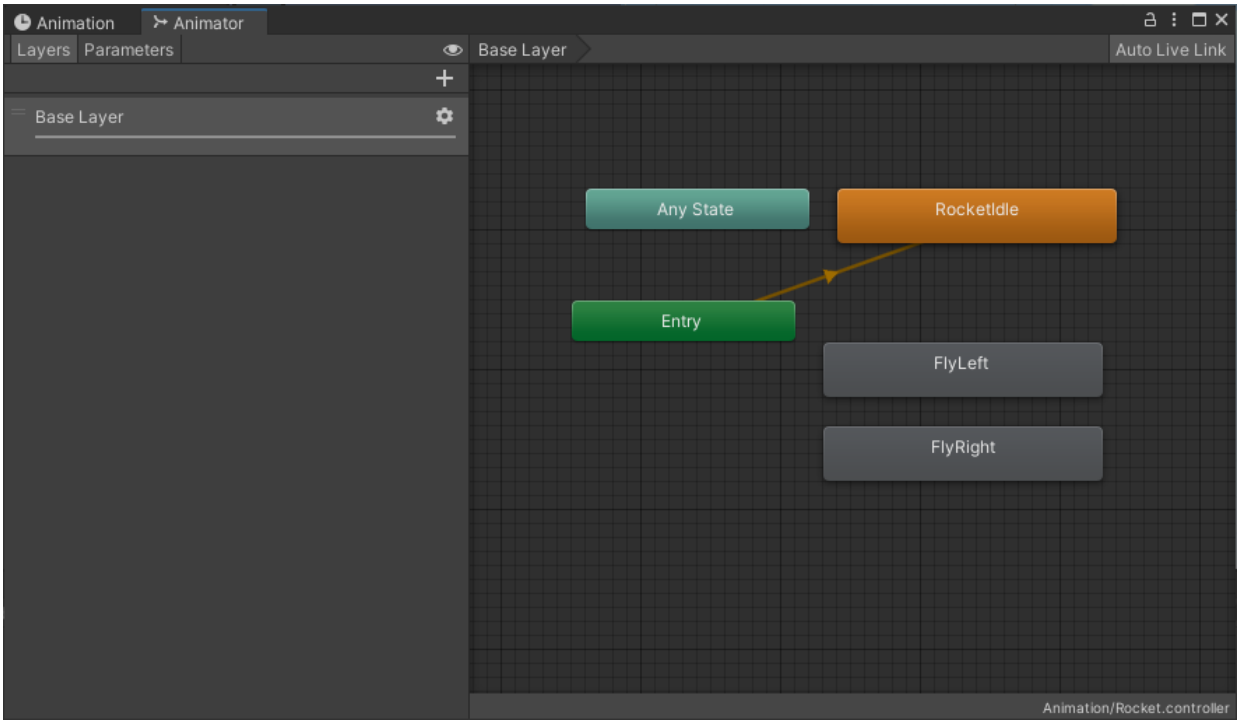
RocketAnimate.cs Script

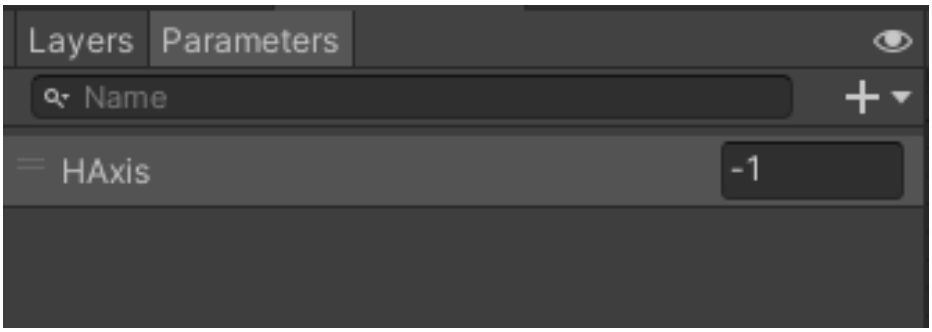
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class RocketAnimation : MonoBehaviour
{
	[Header("Animator")]
	public Animator animator;
	// Start is called before the first frame update
	void Start()
	{
		animator = GetComponent<Animator>();
	}


	// FixedUpdate is called several times per frame
	void FixedUpdate()
	{
		//Get Axis Horizontal gets the Left and Right movement
		//Which is a number from -1 to 1
		float horizontal = Input.GetAxis("Horizontal");
		animator.SetFloat("HAxis", horizontal);
	}
}
```

Rocket Animation Frames

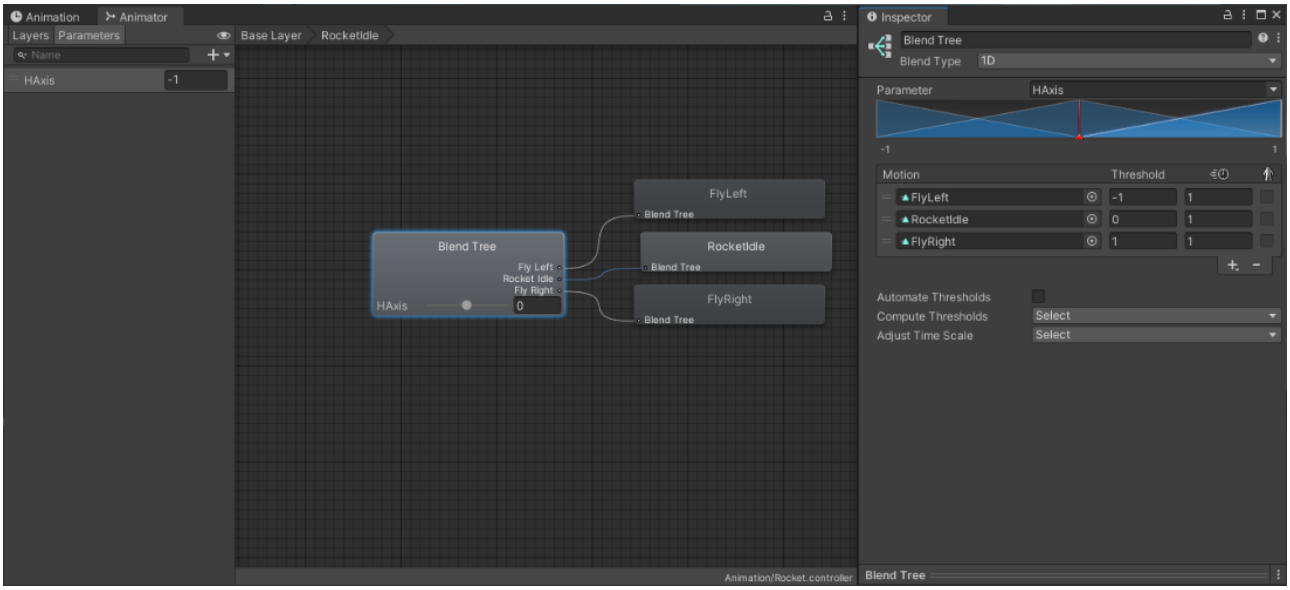


Rocket Animator Tree

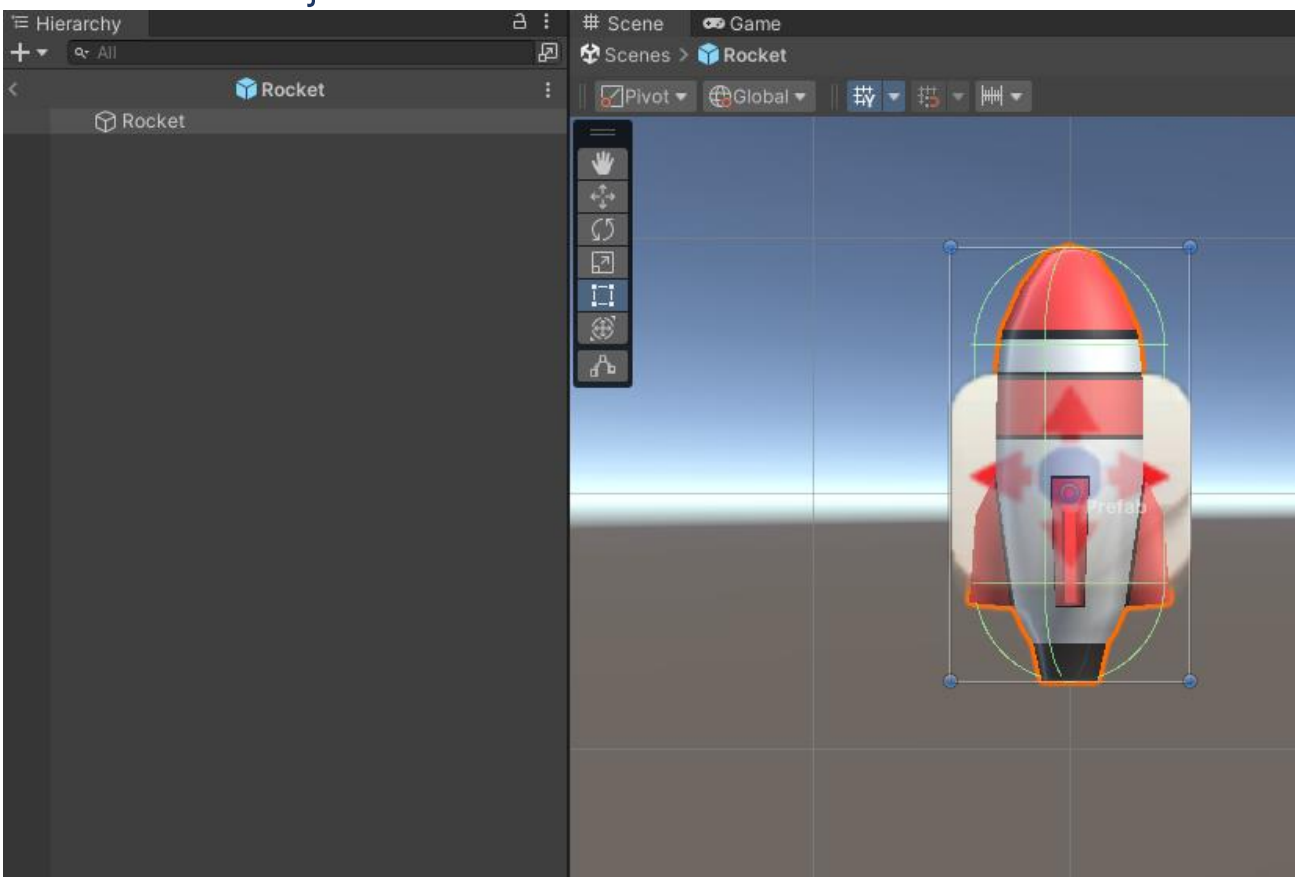




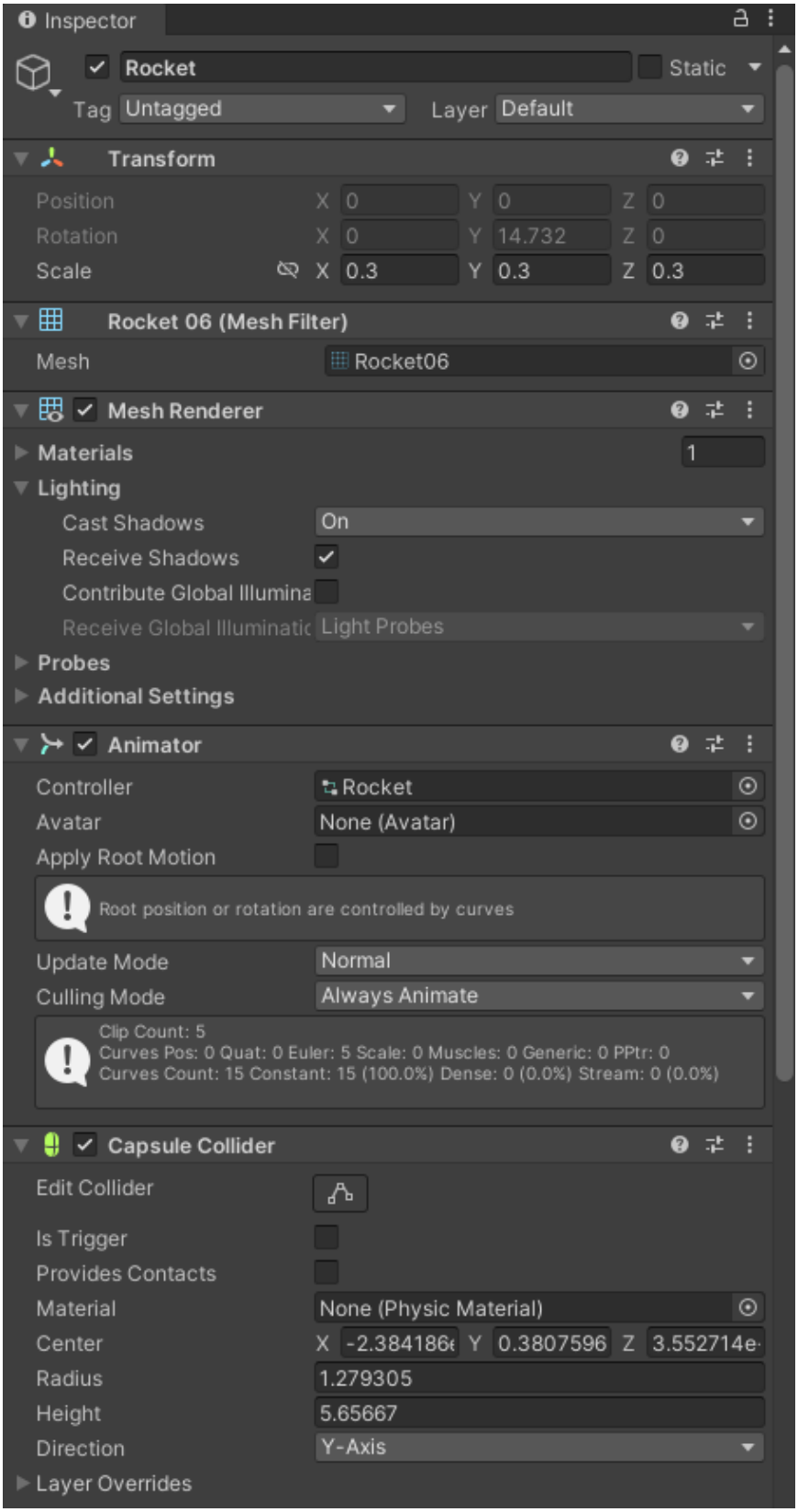
Rocket Blend Tree



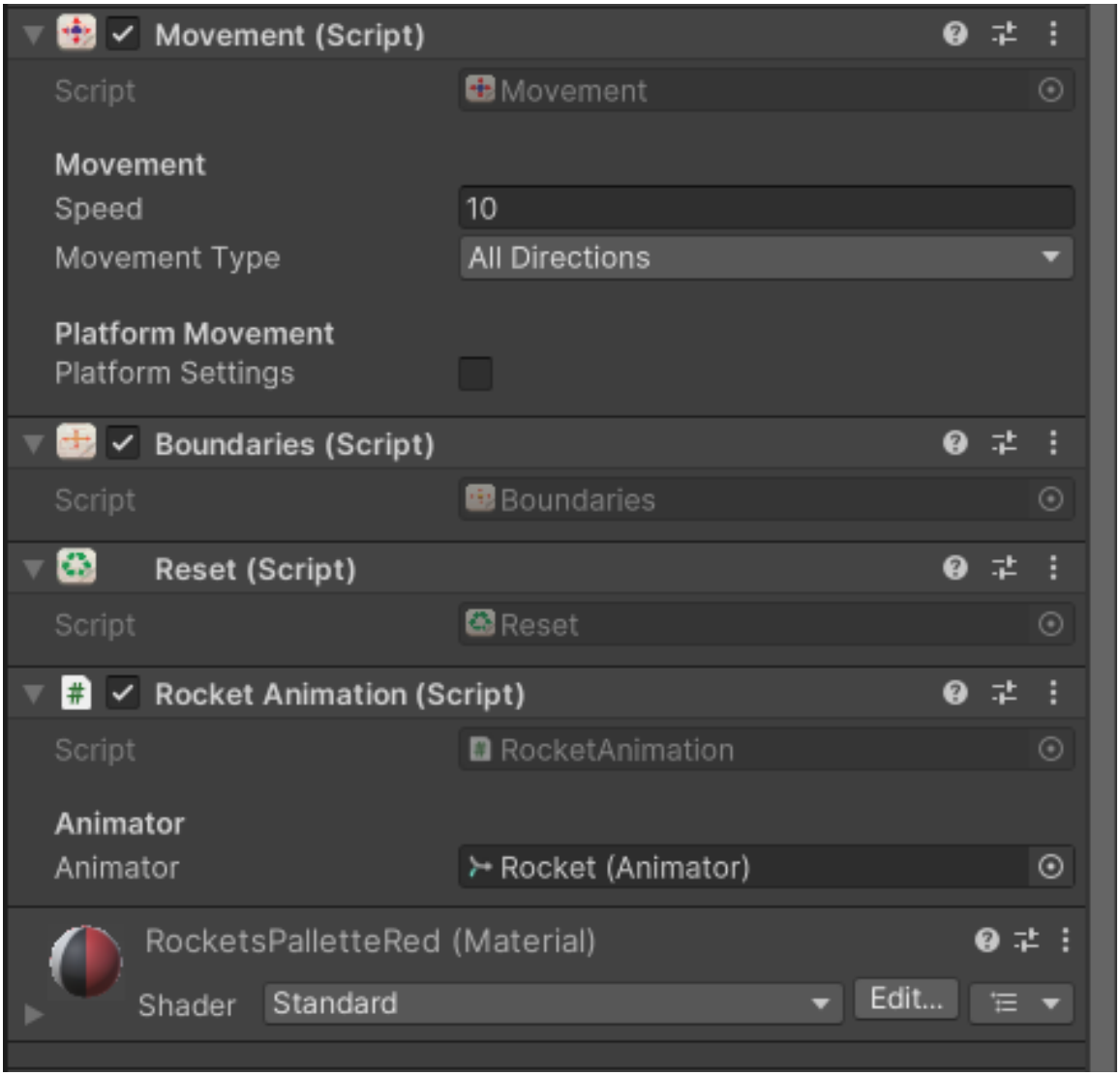
Rocket Prefab Object



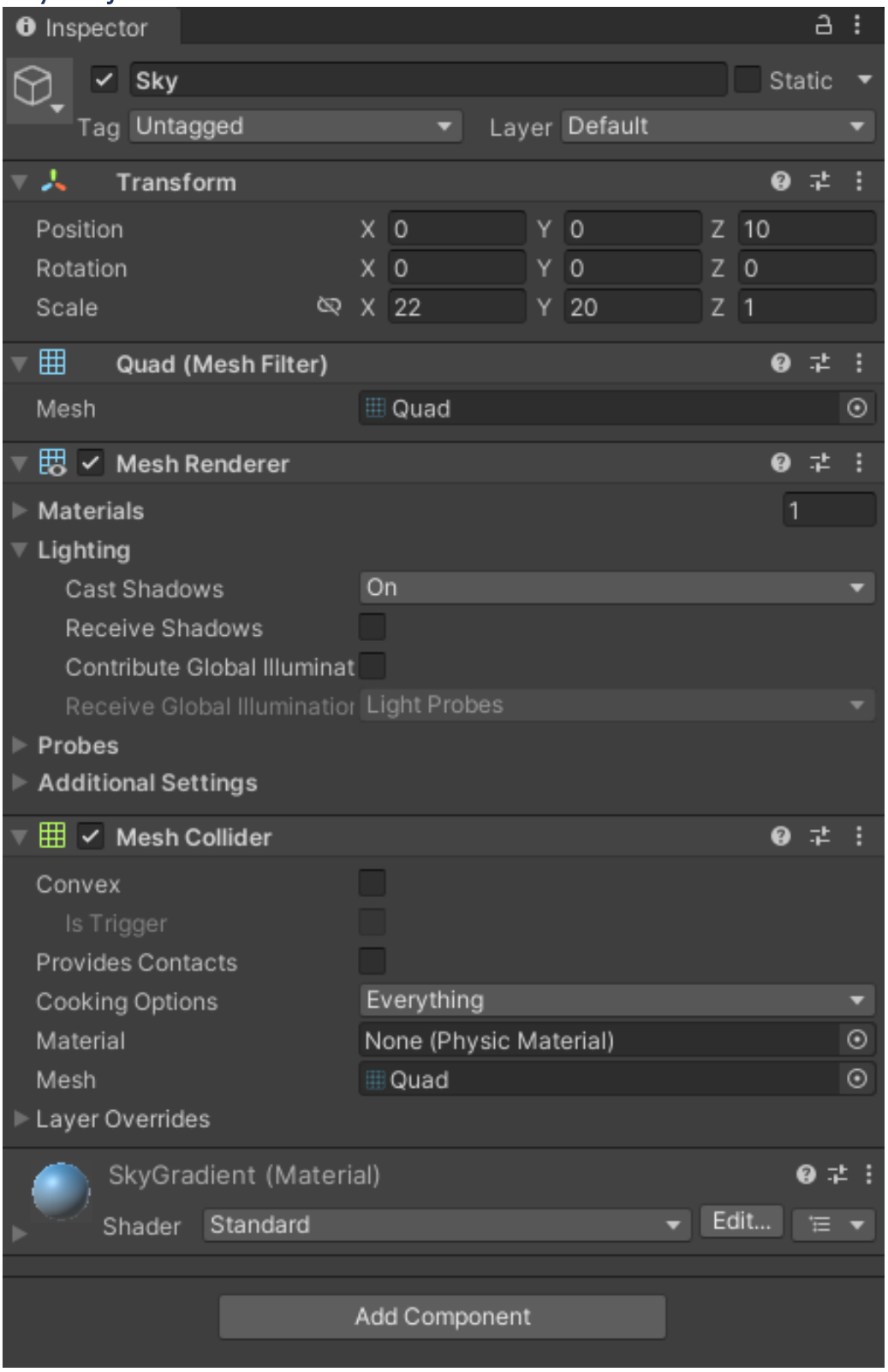
Rocket Prefab Object Continued



Rocket Prefab Object Continued

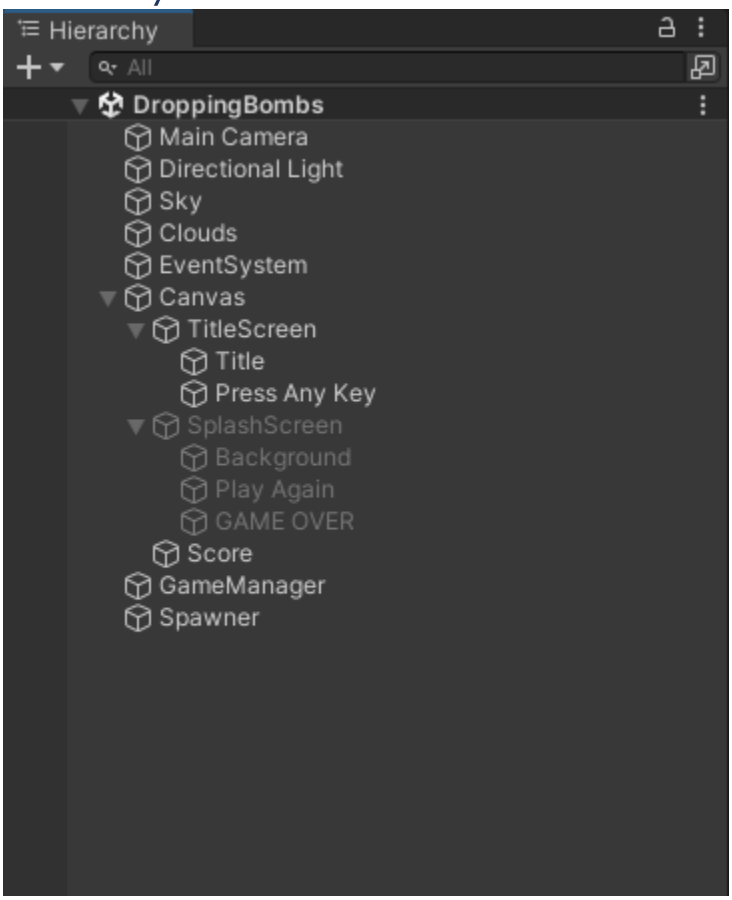


Sky Object



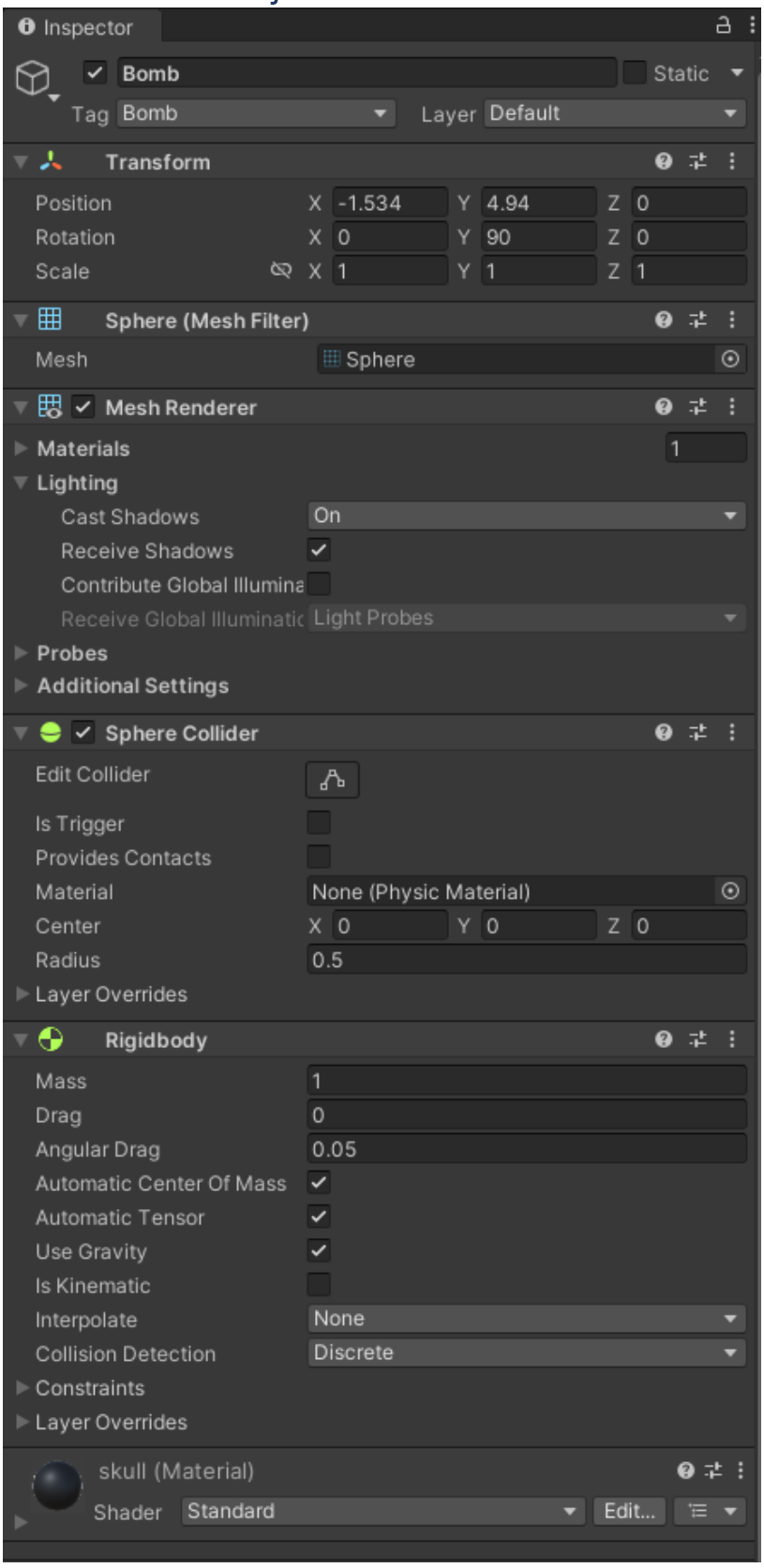
Activity Solution: Dropping Bombs Part 3

Hierarchy

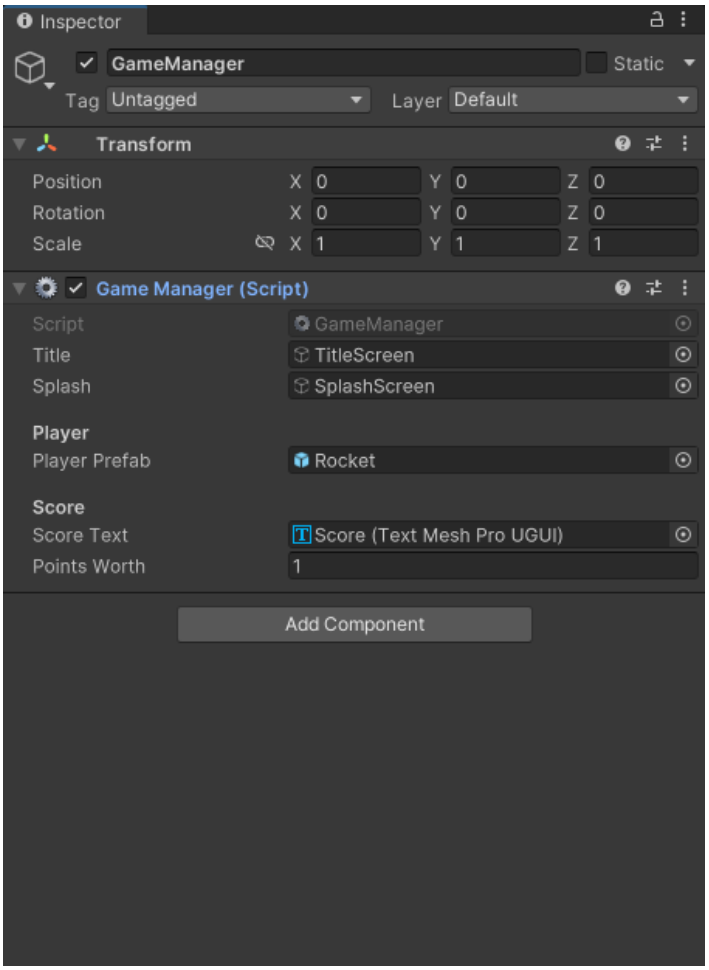


Background Object

Bomb Prefab Object



GameManager Object

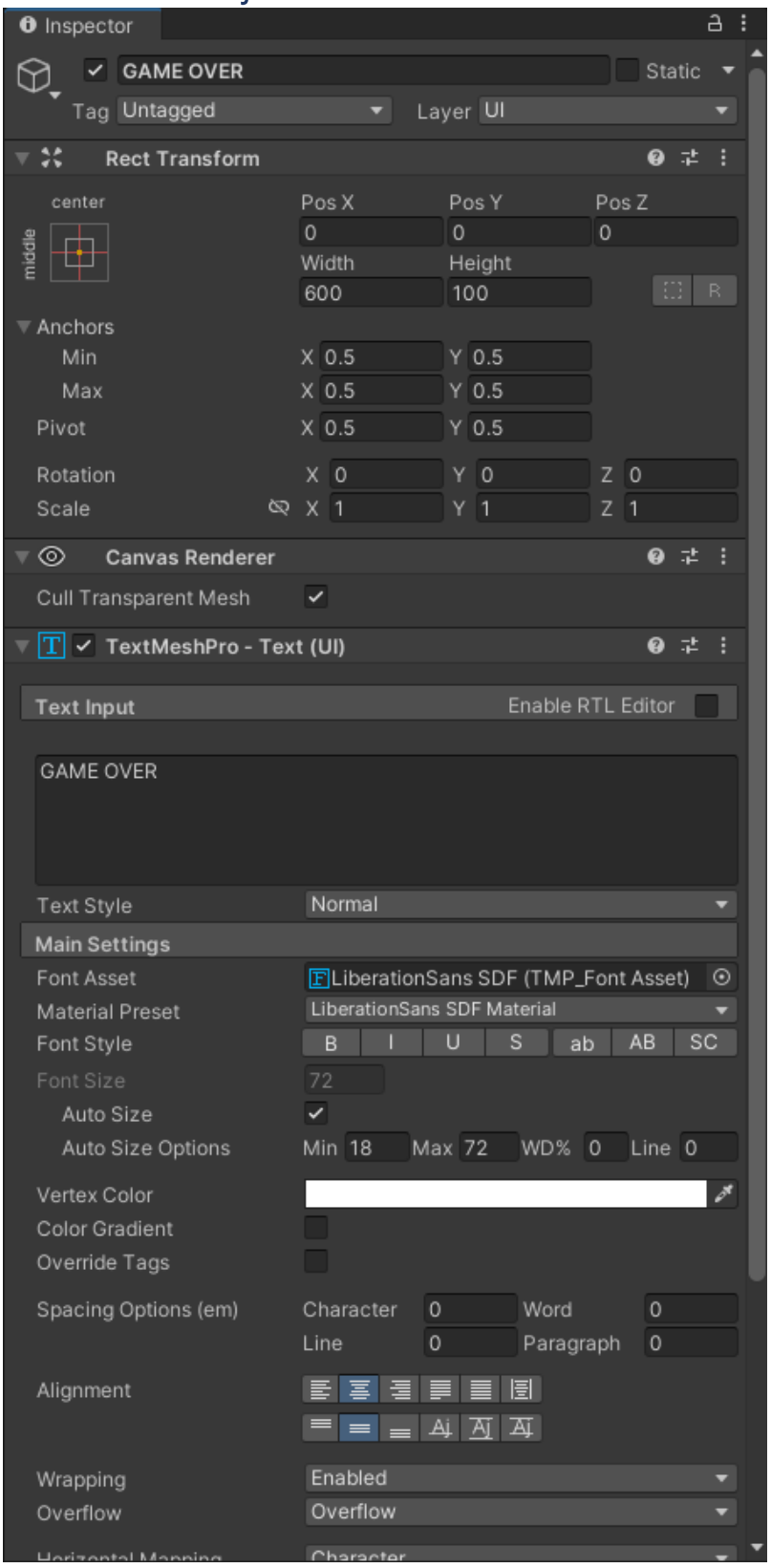


GameManager.cs Script

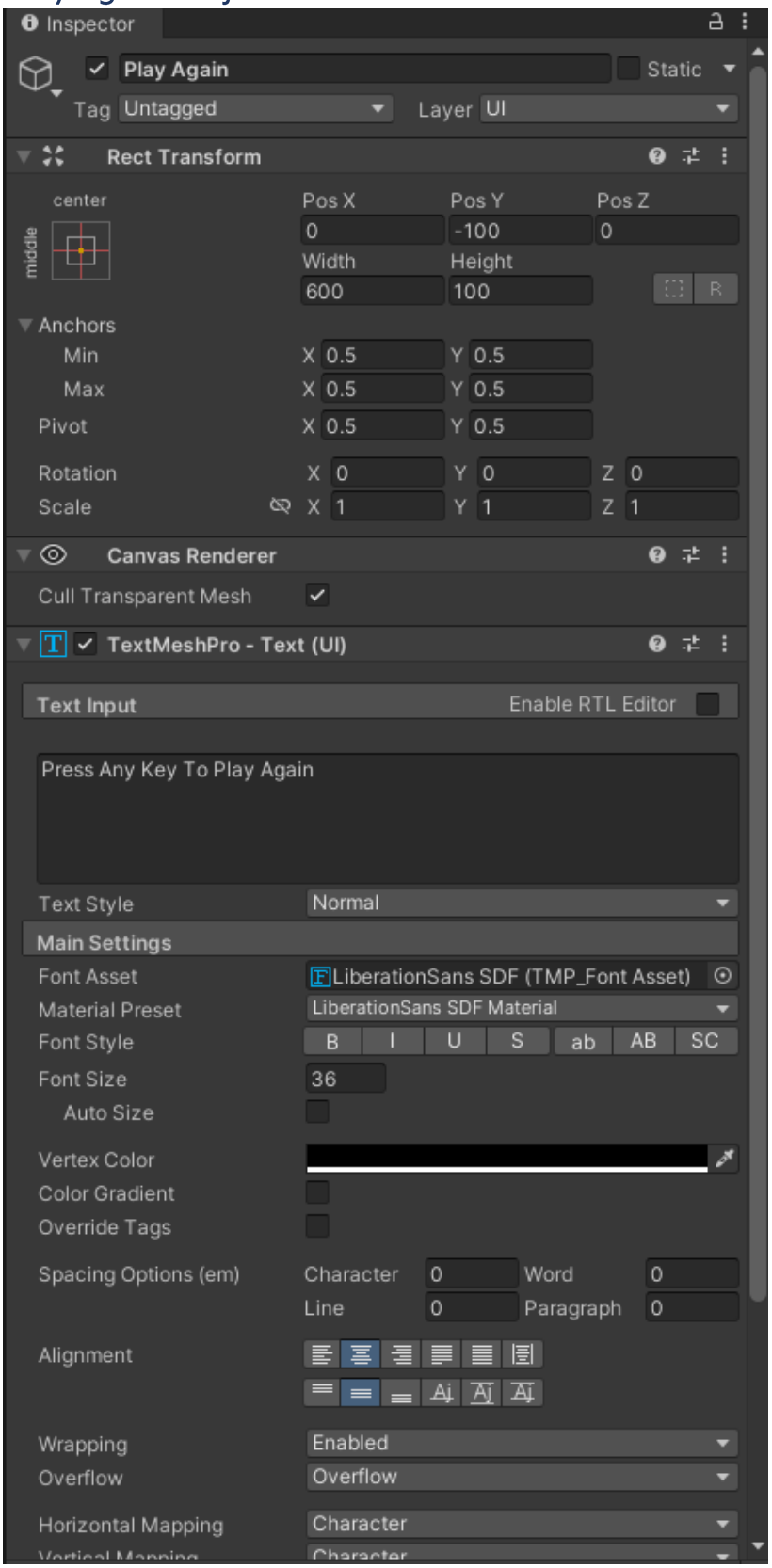
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;

public class GameManager : MonoBehaviour
{
	private Spawner spawner;
	public GameObject title;
	private Vector2 screenBounds;


	public GameObject splash;
	[Header("Player")]
	public GameObject playerPrefab;
	private GameObject player;
	private bool gameStarted = false;

	[Header("Score")]
	public TMP_Text scoreText;
	public int pointsWorth = 1;
	private int score;

	private void Awake()
	{
		spawner = GameObject.Find("Spawner").GetComponent<Spawner>();
		screenBounds = Camera.main.ScreenToWorldPoint(new Vector3(Screen.width,
		Screen.height, Camera.main.transform.position.z));
		scoreText.enabled = false;
	}

	// Start is called before the first frame update
	void Start()
	{
		spawner.active = false;
		title.SetActive(true);
		splash.SetActive(false);
	}


	// Update is called once per frame
	void Update()
	{
		if (!gameStarted)
		{
			if (Input.anyKeyDown)
			{
				ResetGame();
			}
		}
		else
		{
			if (!player)
			{
				OnPlayerKilled();
```

```
                }
        }

        var nextBomb = GameObject.FindGameObjectsWithTag("Bomb");

                foreach (GameObject bombObject in nextBomb)
                {
                        if (bombObject.transform.position.y < (-screenBounds.y - 12))
                        {
                                if (gameStarted)
                                {
                                score += pointsWorth;
                                scoreText.text = "Score: " + score.ToString();
                                }

                                Destroy(bombObject);
                        }
                }
        }


        void OnPlayerKilled()
        {
                spawner.active = false;
                gameStarted = false;

                splash.SetActive(true);
        }

        void ResetGame()
        {

                spawner.active = true;
                title.SetActive(false);

                splash.SetActive(false);

                scoreText.enabled = true;
                score = 0;
                scoreText.text = "Score: " + score.ToString();

                player = Instantiate(playerPrefab,new Vector3(0,0,0),
                playerPrefab.transform.rotation);
                gameStarted = true;
        }
}
```

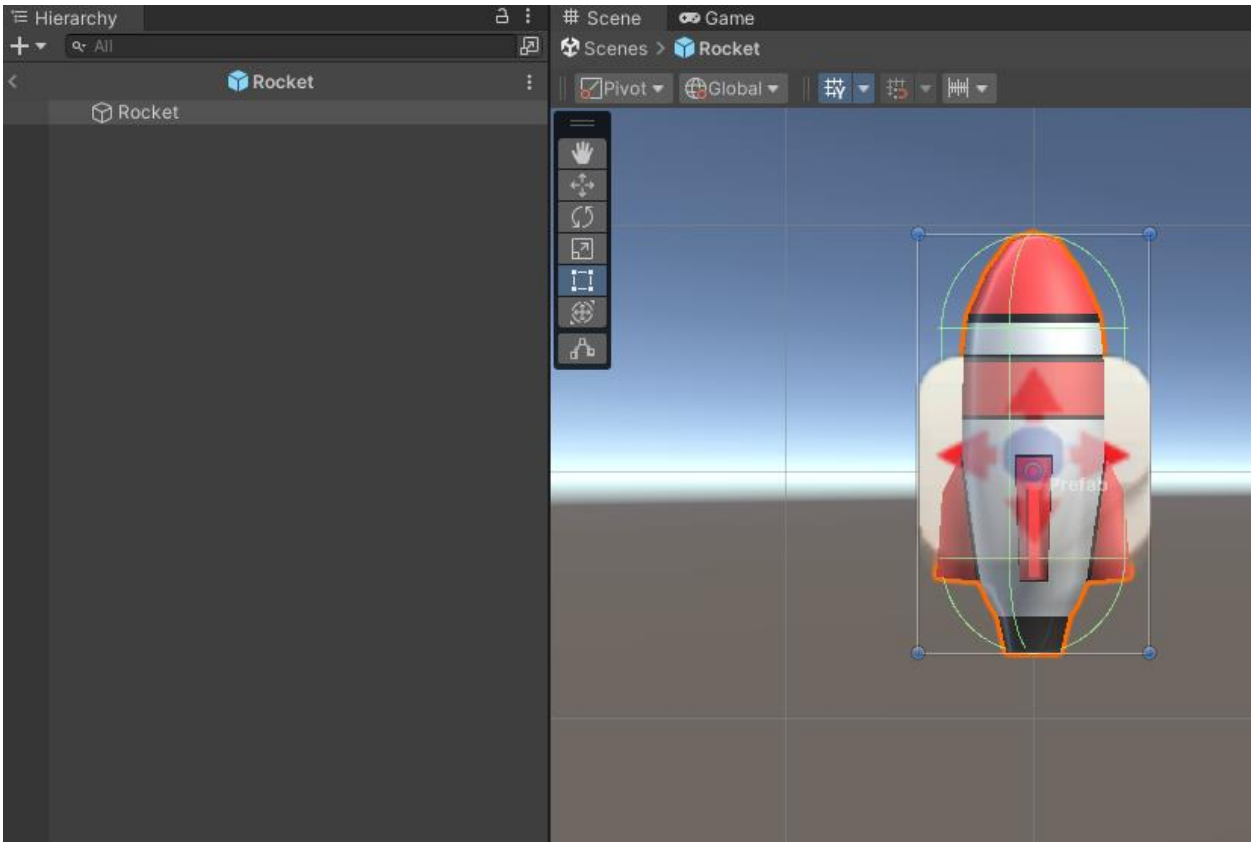
GameOver Object

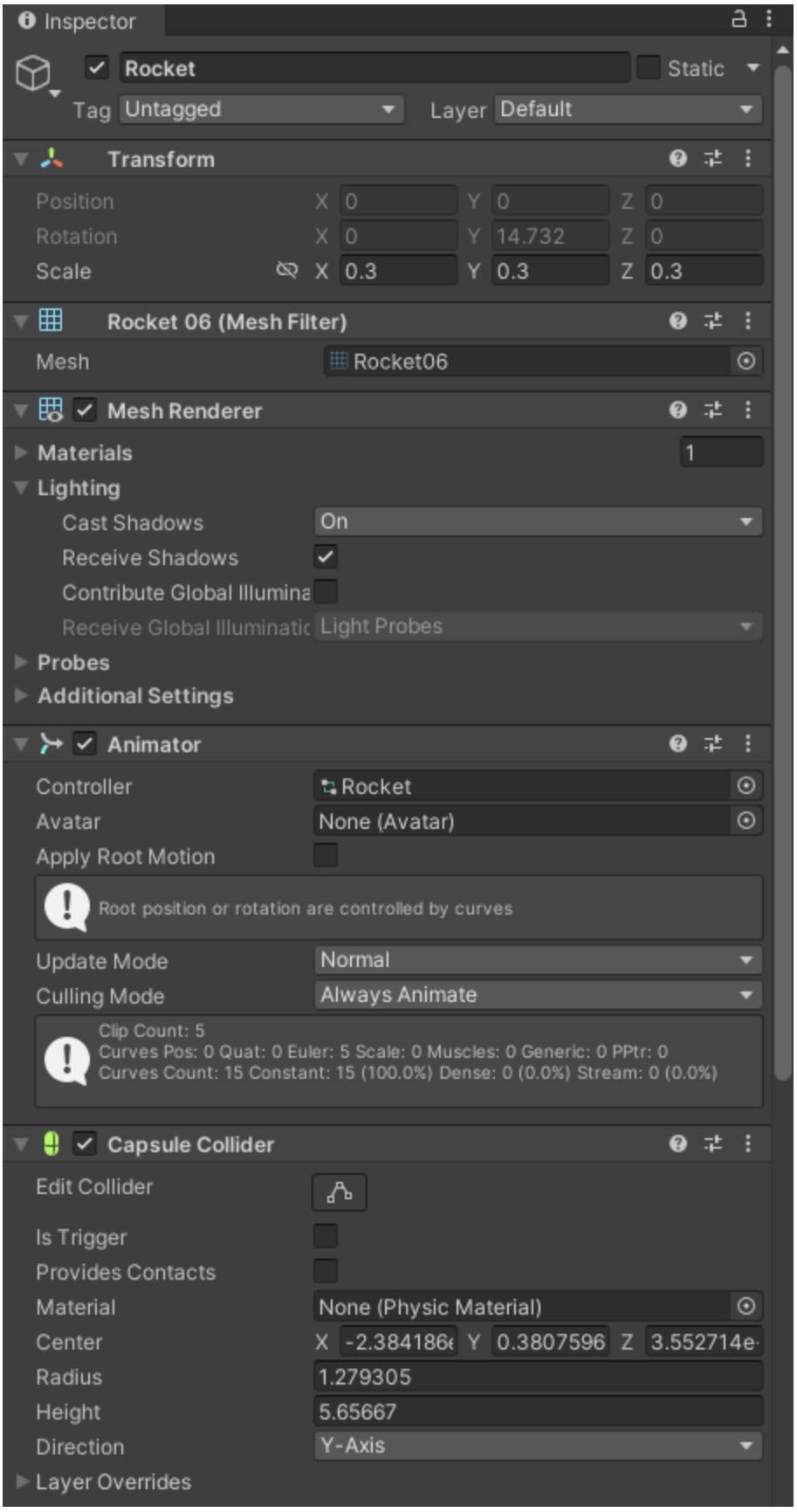


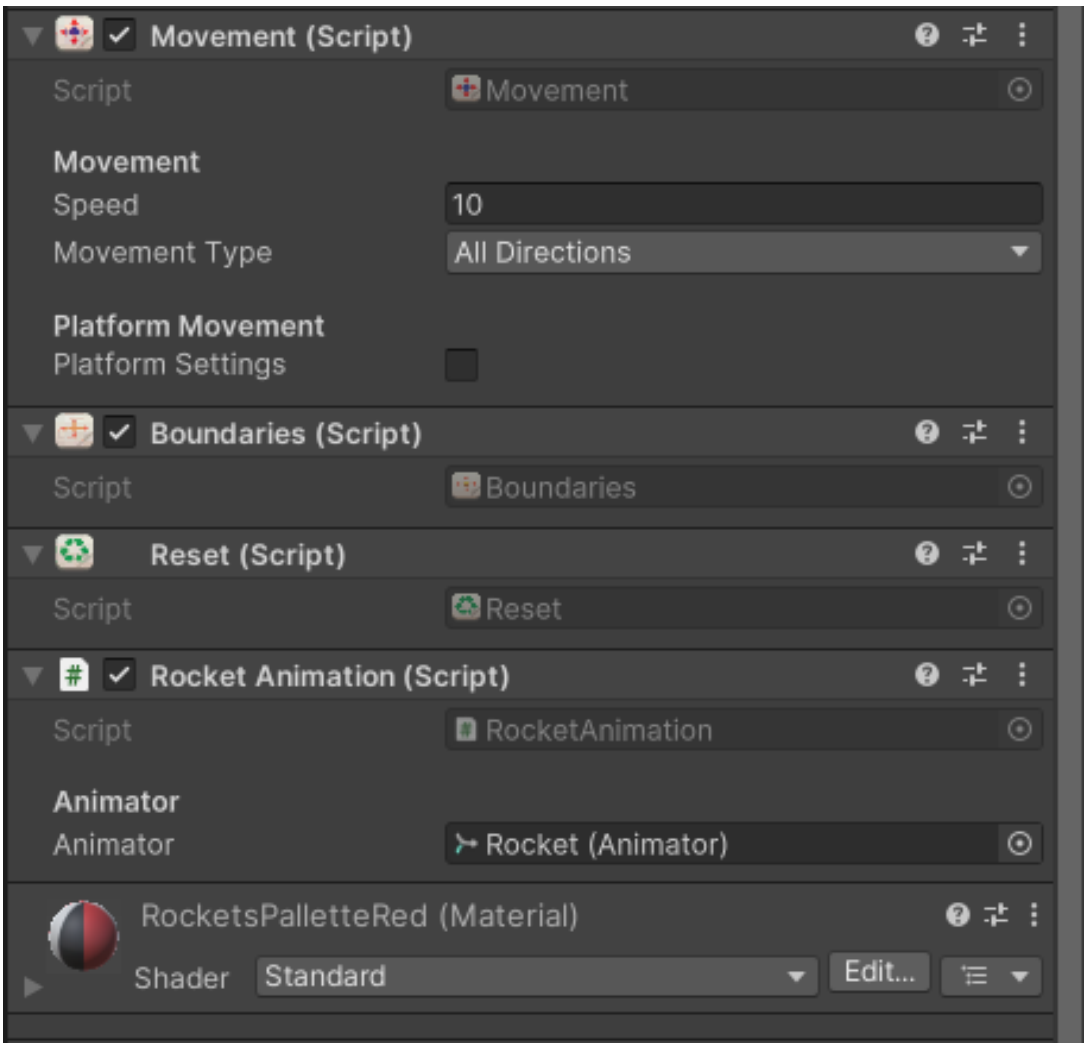
PlayAgain Object



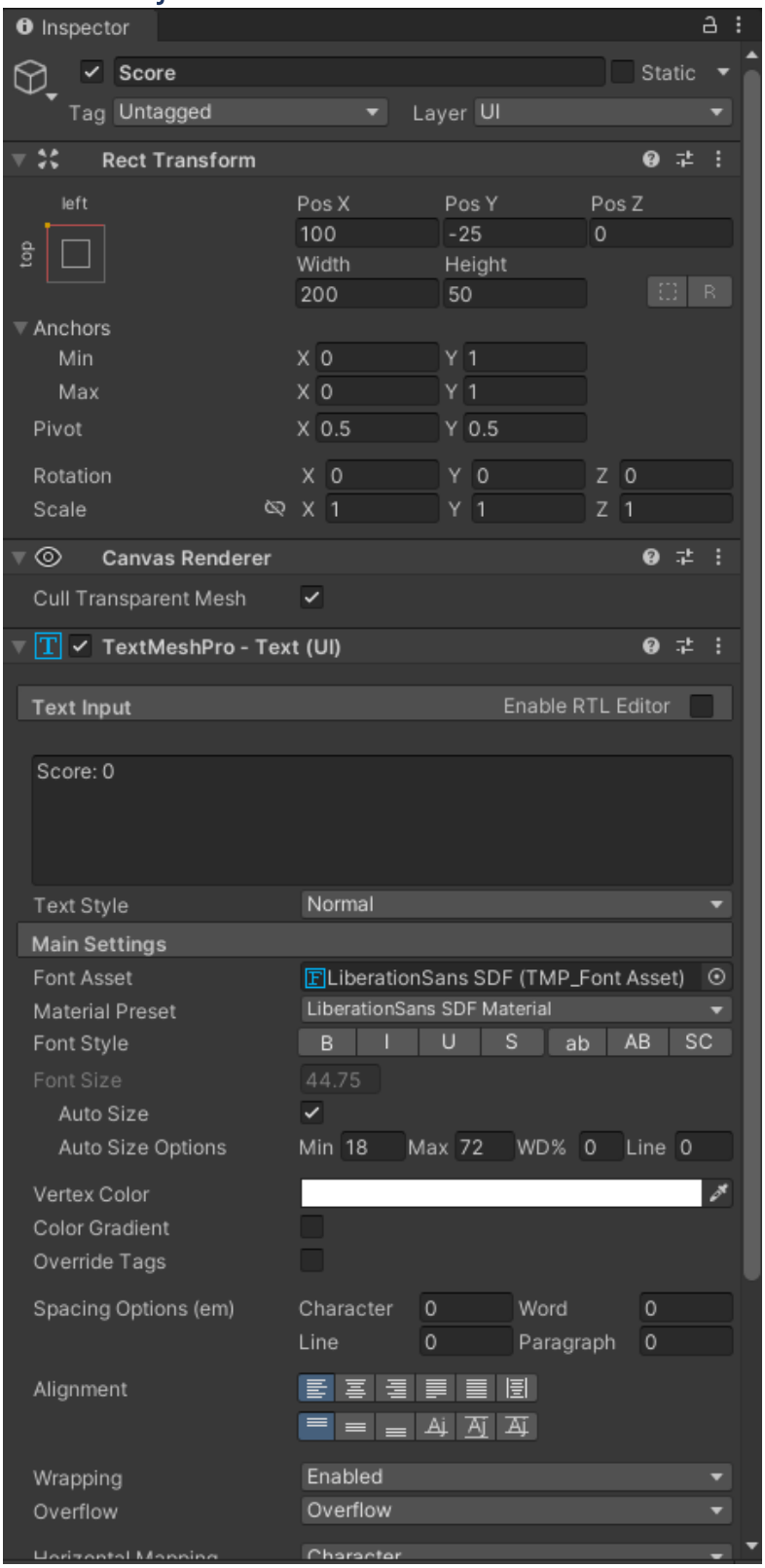
Rocket Prefab Object



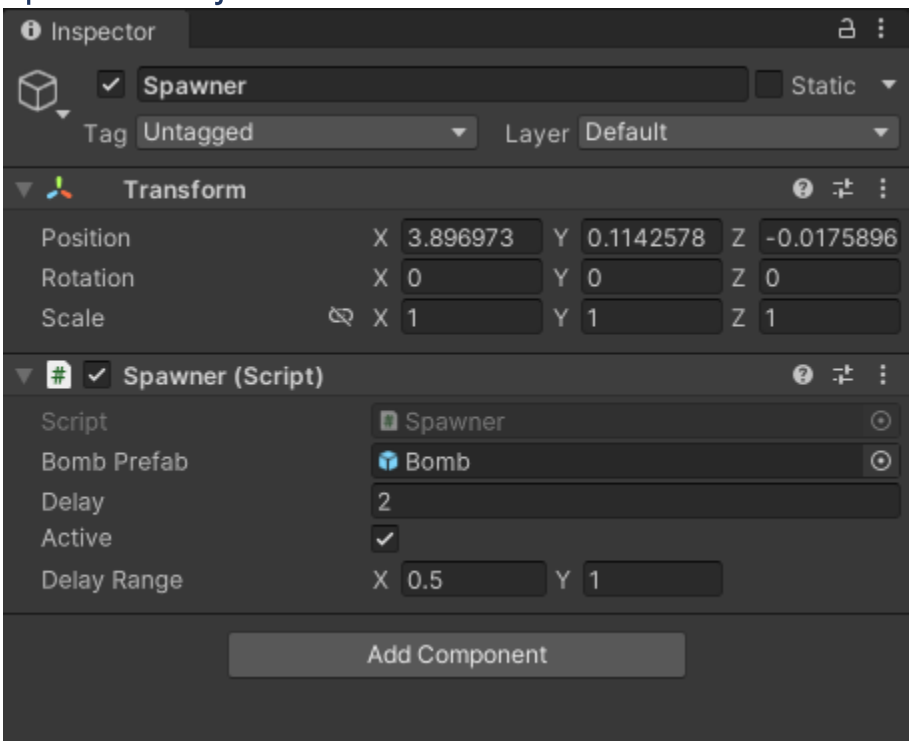




Score Object



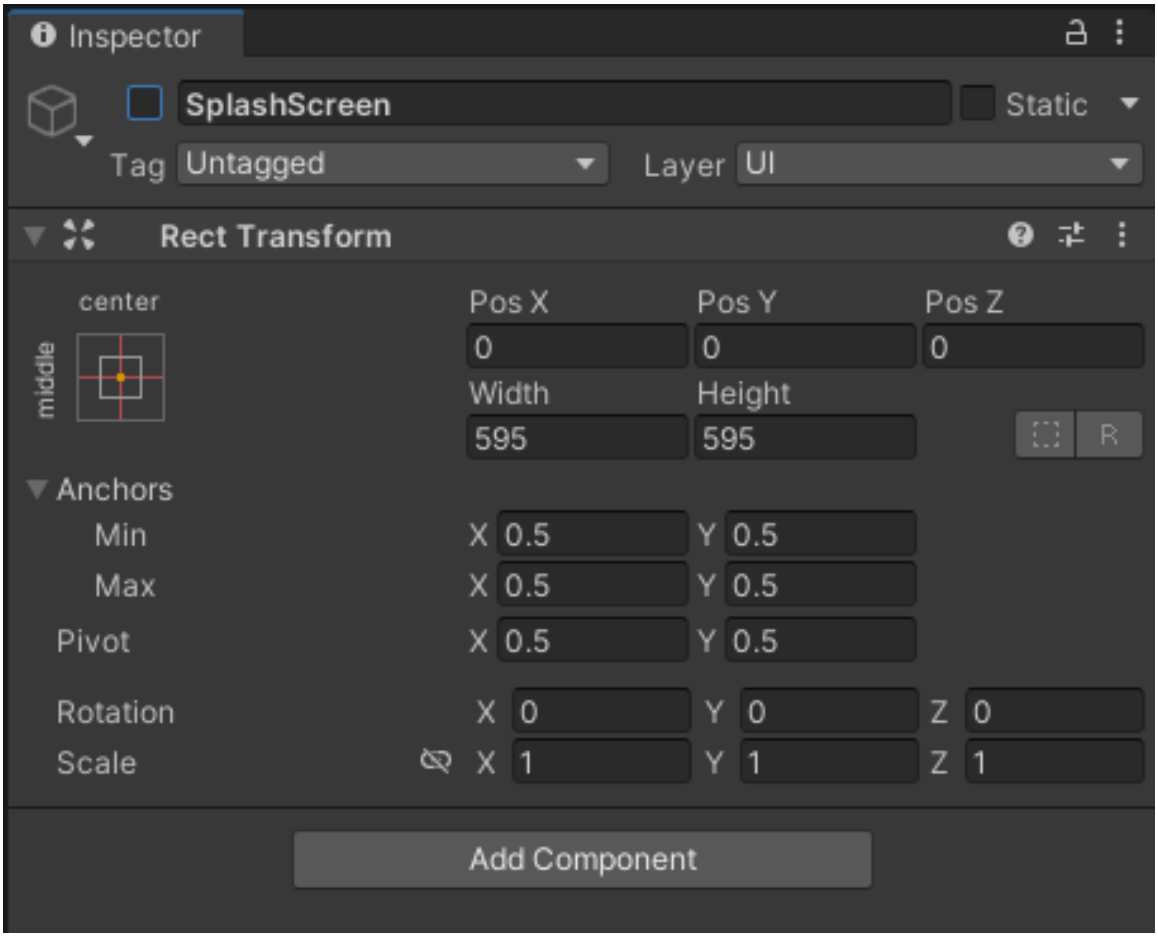
Spawner Object



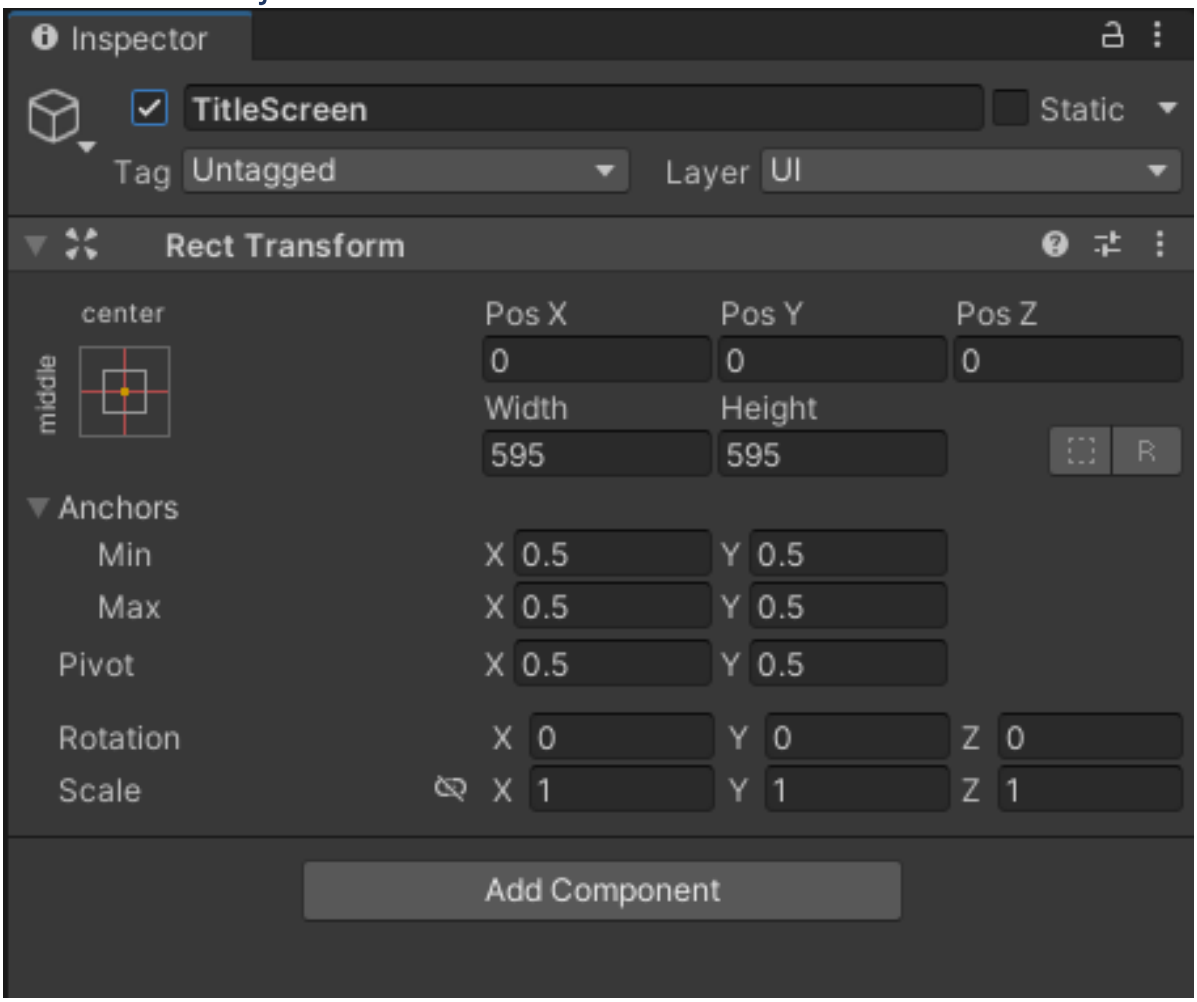
Spawner.cs Script

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

public class Spawner : MonoBehaviour
{
	public GameObject bombPrefab;
	public float delay = 2.0f;
	public bool active = true;
	public Vector2 delayRange = new Vector2(1, 2);
	private Vector2 screenBounds;

	// Start is called before the first frame update
	void Start()
	{
		ResetDelay();
		StartCoroutine(EnemyGenerator());


		screenBounds = Camera.main.ScreenToWorldPoint(new Vector3(Screen.width,
		Screen.height, Camera.main.transform.position.z));
	}

	IEnumerator EnemyGenerator()
	{
		yield return new WaitForSeconds(delay);
		if(active) {
		float randomX = Random.Range(-screenBounds.x, screenBounds.x);
		float spawnY = screenBounds.y + 1;


		Instantiate(bombPrefab, new Vector3(randomX, spawnY, 0),
		bombPrefab.transform.rotation);
		ResetDelay();
	}

		StartCoroutine(EnemyGenerator());
	}


	void ResetDelay()
	{
		delay = Random.Range(delayRange.x, delayRange.y);
	}
}
```

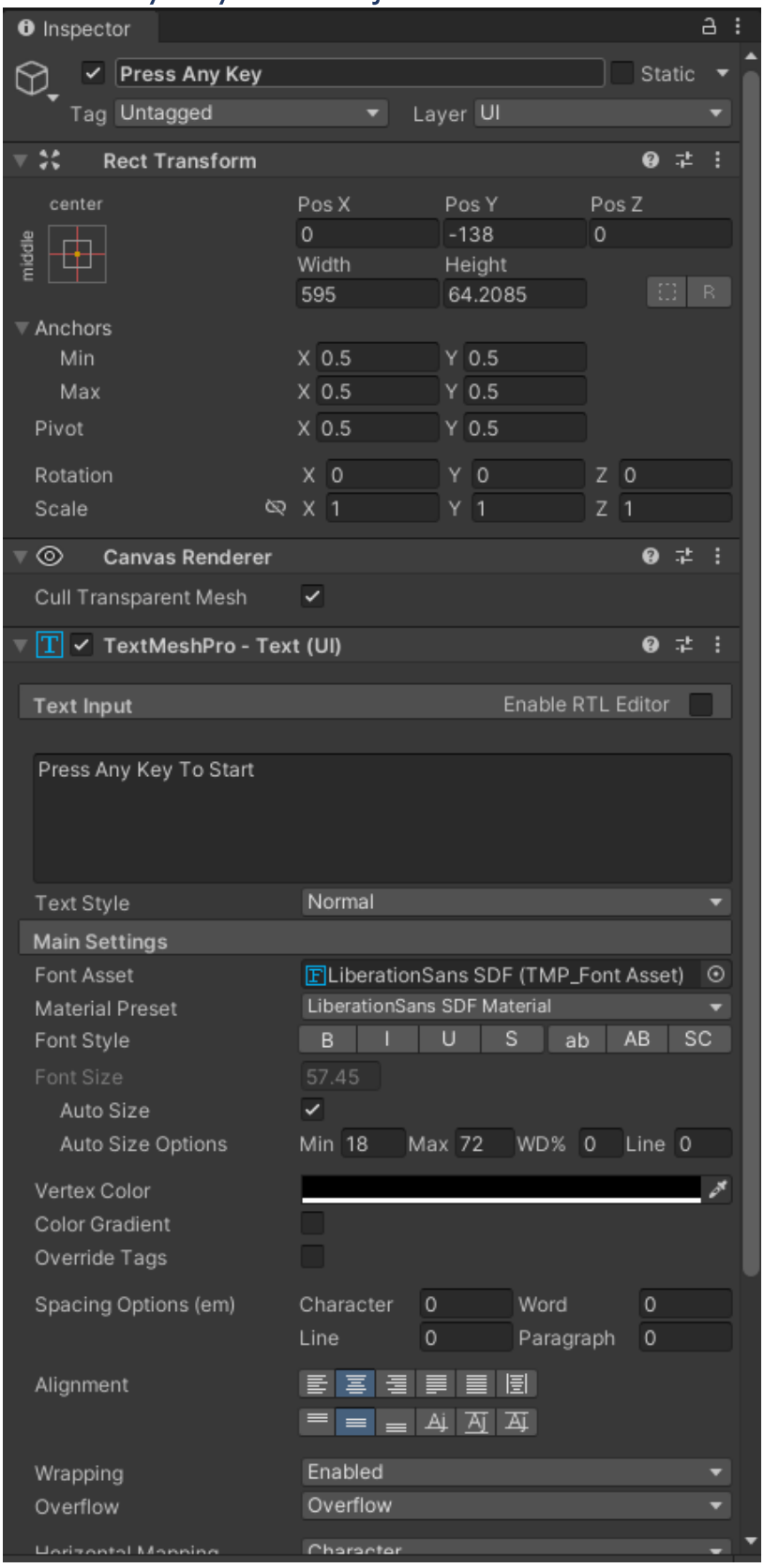
SplashScreen Object



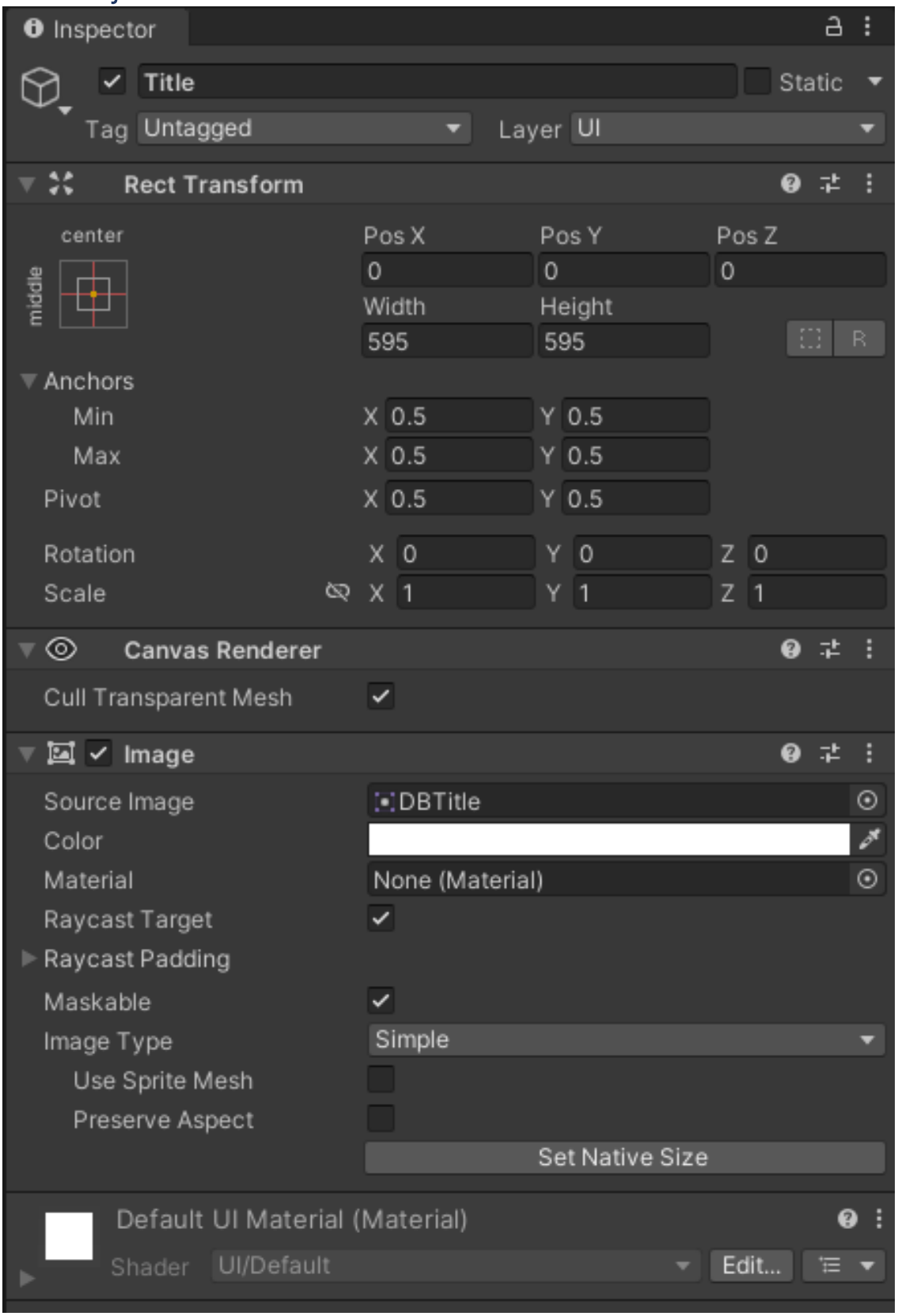
TitleScreen Object



Press Any Key Text Object

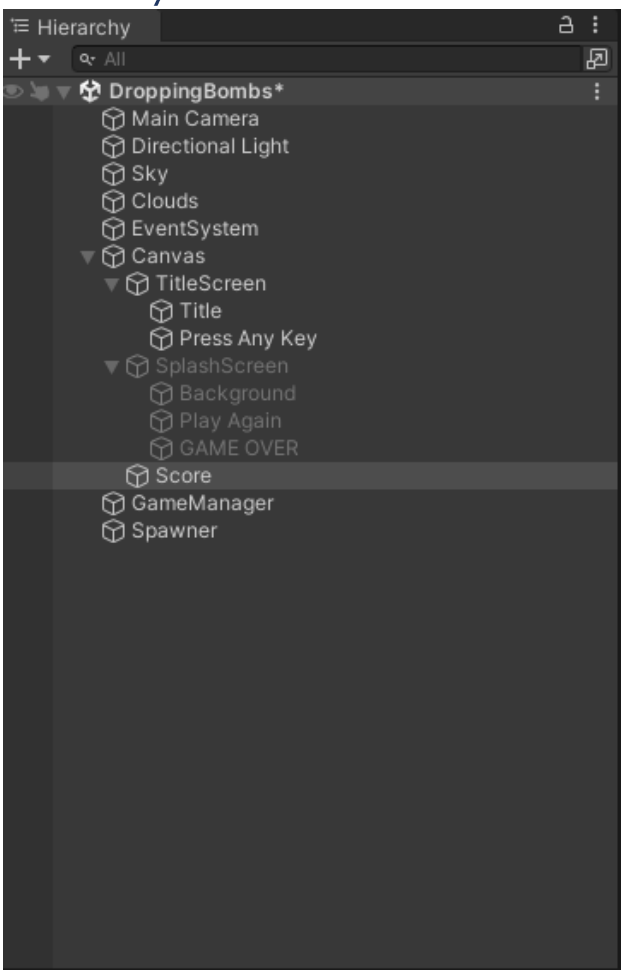


Title Object



Activity Solution: Dropping Bombs Part 4

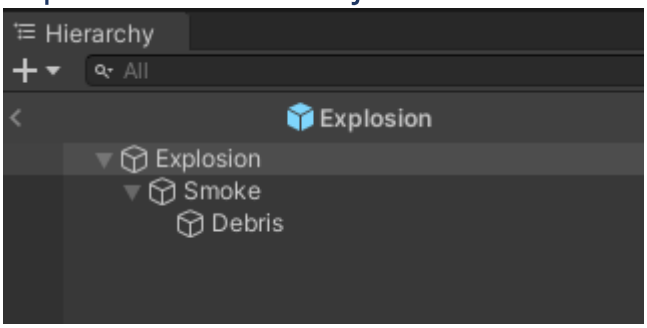
Hierarchy

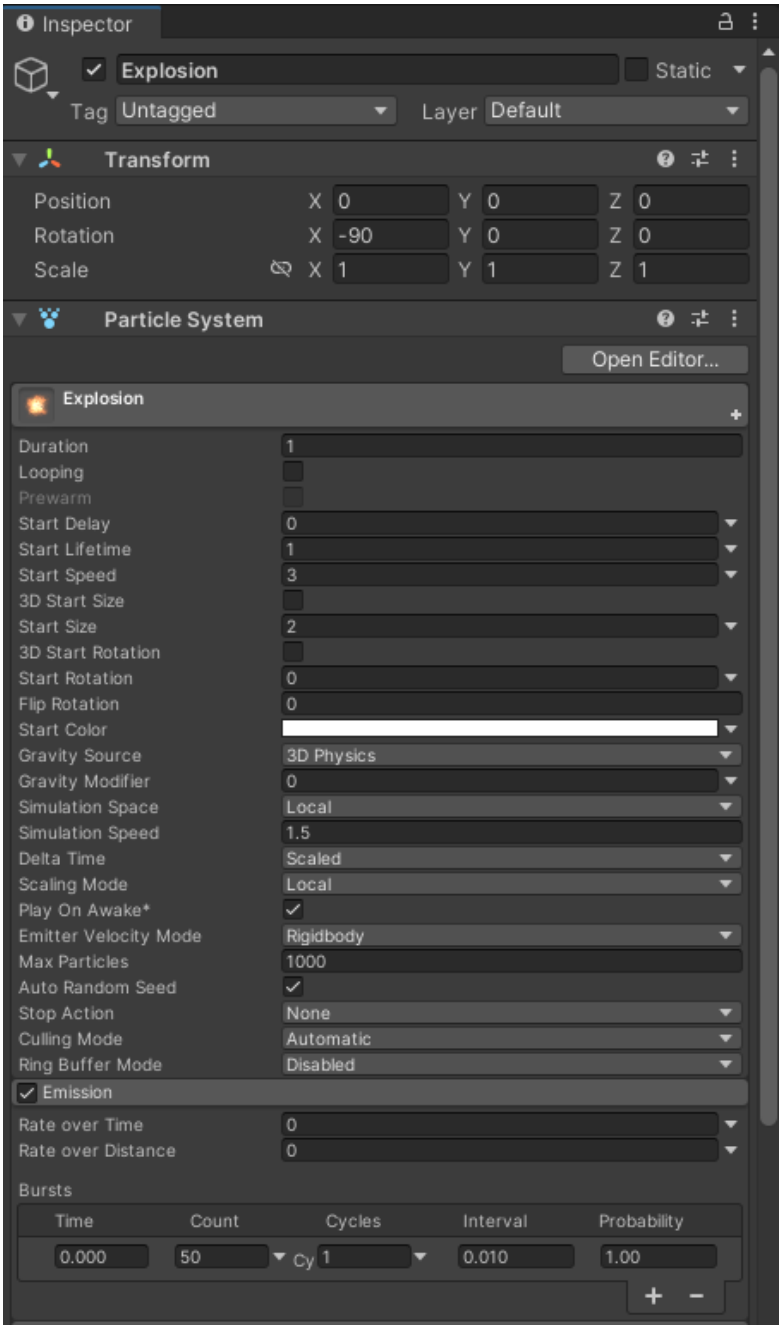


ExplosionClear.cs Script

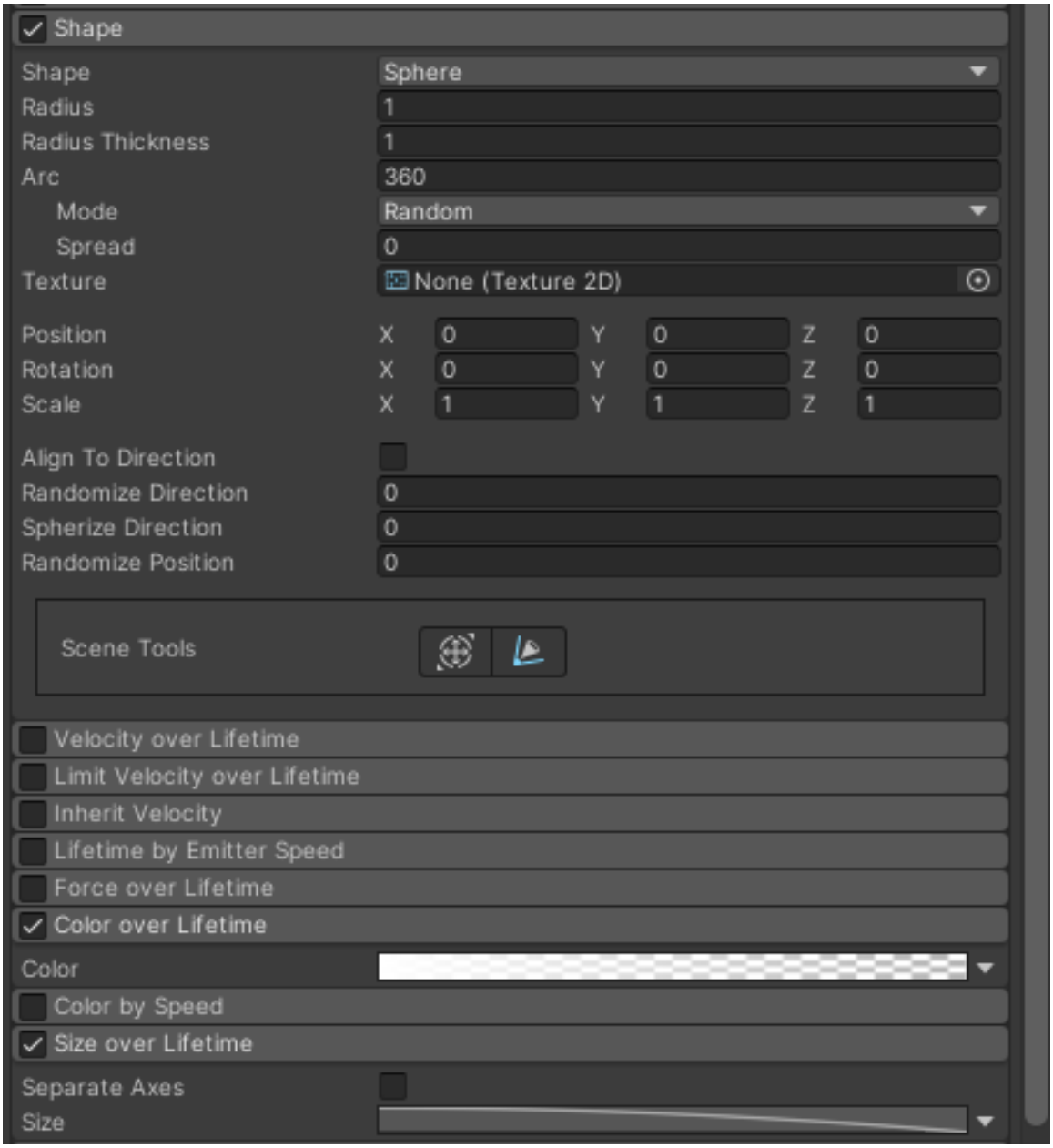
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ExplosionClear : MonoBehaviour
{
	private ParticleSystem particleSmoke;


	private void Awake()
	{
		particleSmoke = GetComponentInChildren<ParticleSystem>();
	}


	private void Update()
	{
		if (!particleSmoke.IsAlive())
		{
			Destroy(gameObject);
		}
	}
}
```

Explosion Prefab Object

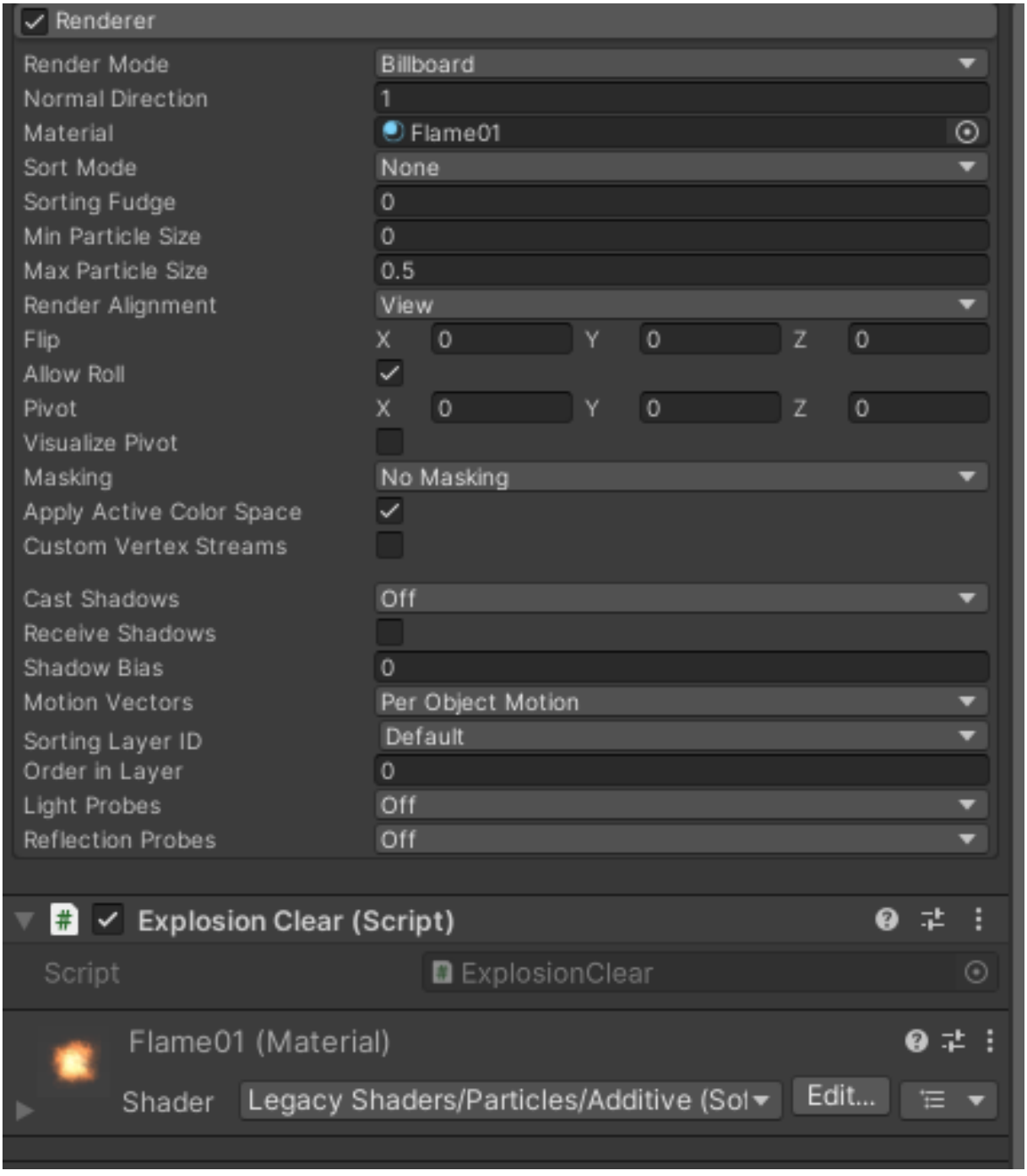




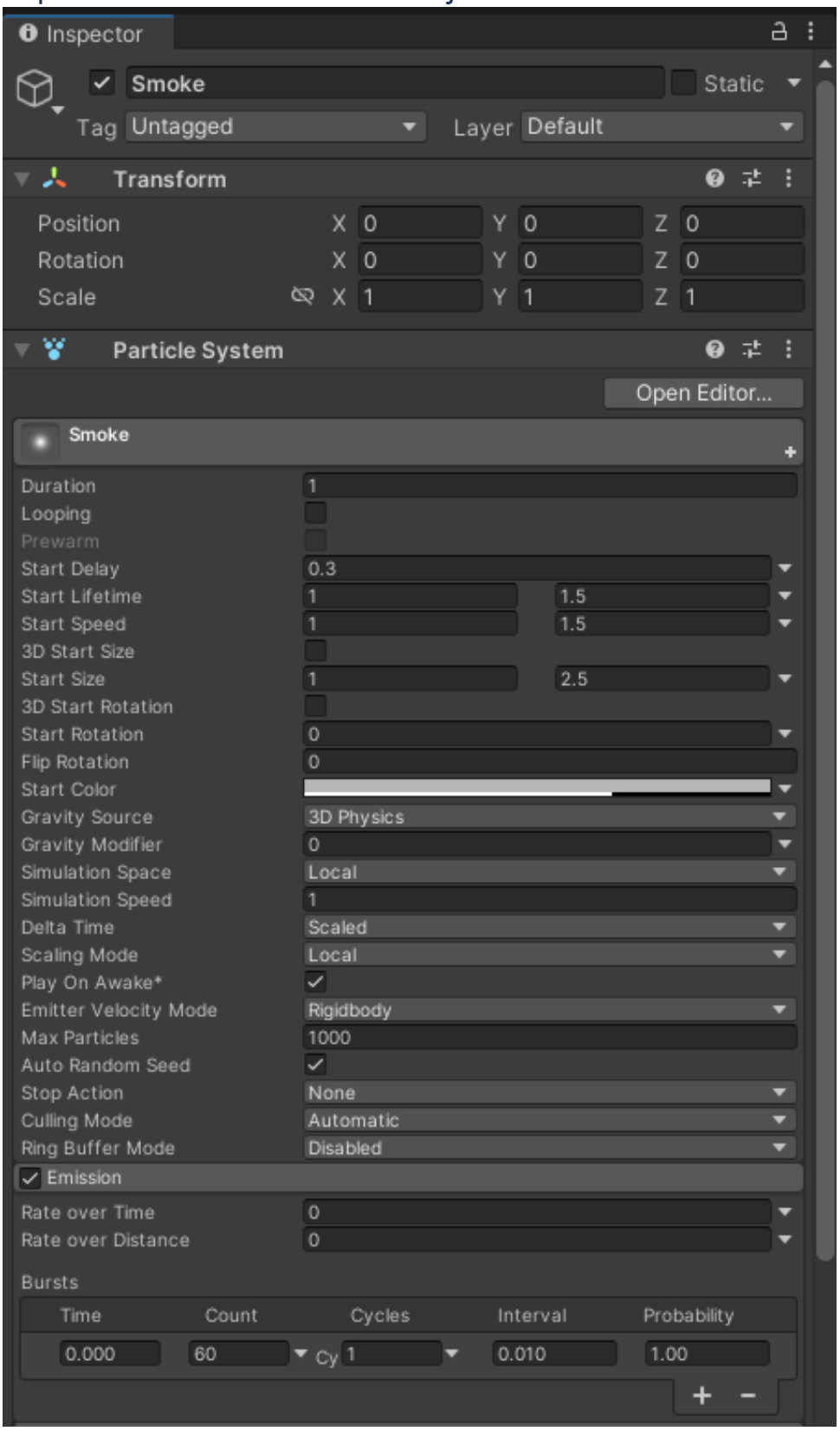
Explosion Prefab Object Continued

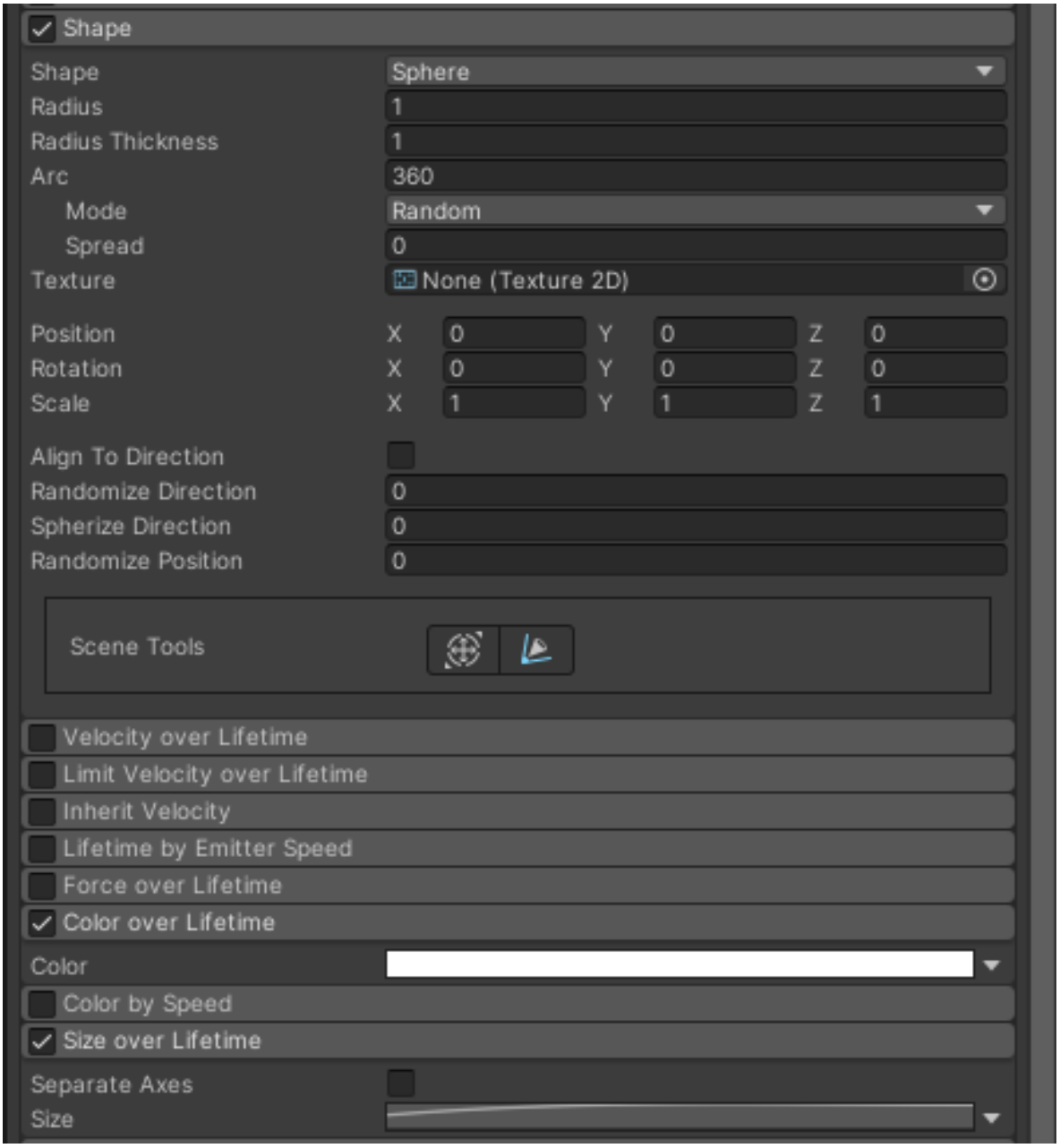


Explosion Prefab Object Continued

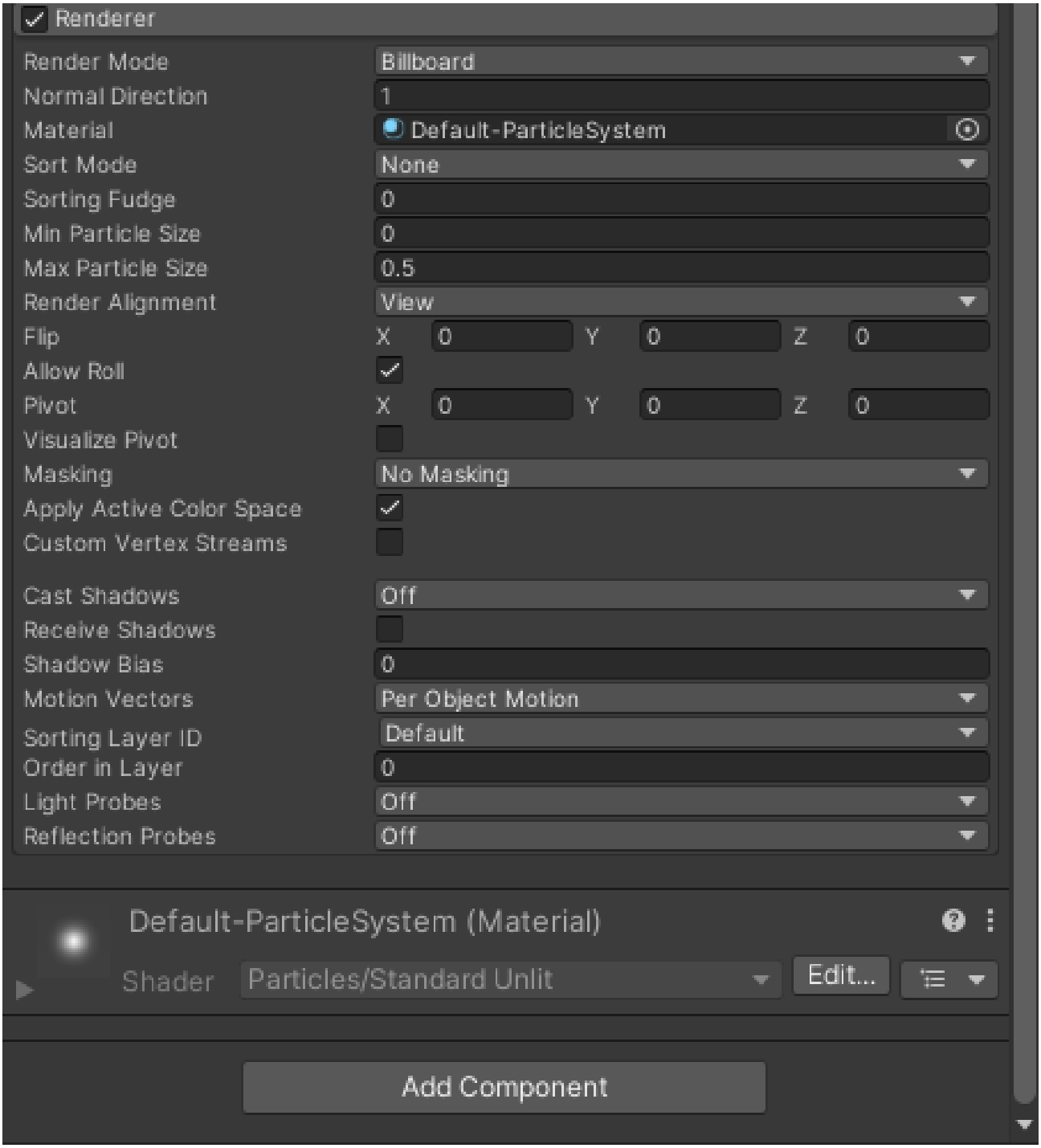


Explosion Prefab > Smoke Object

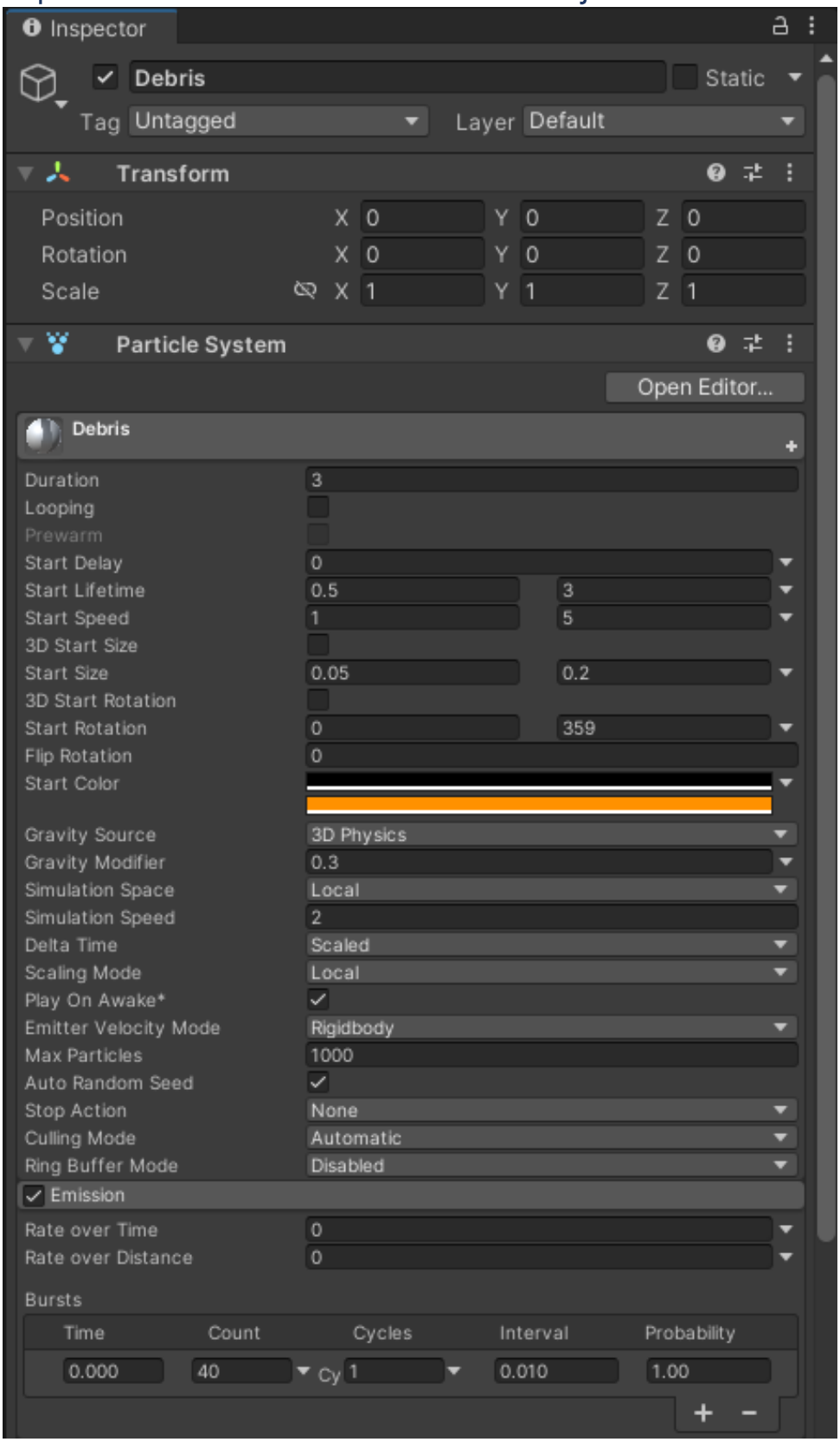


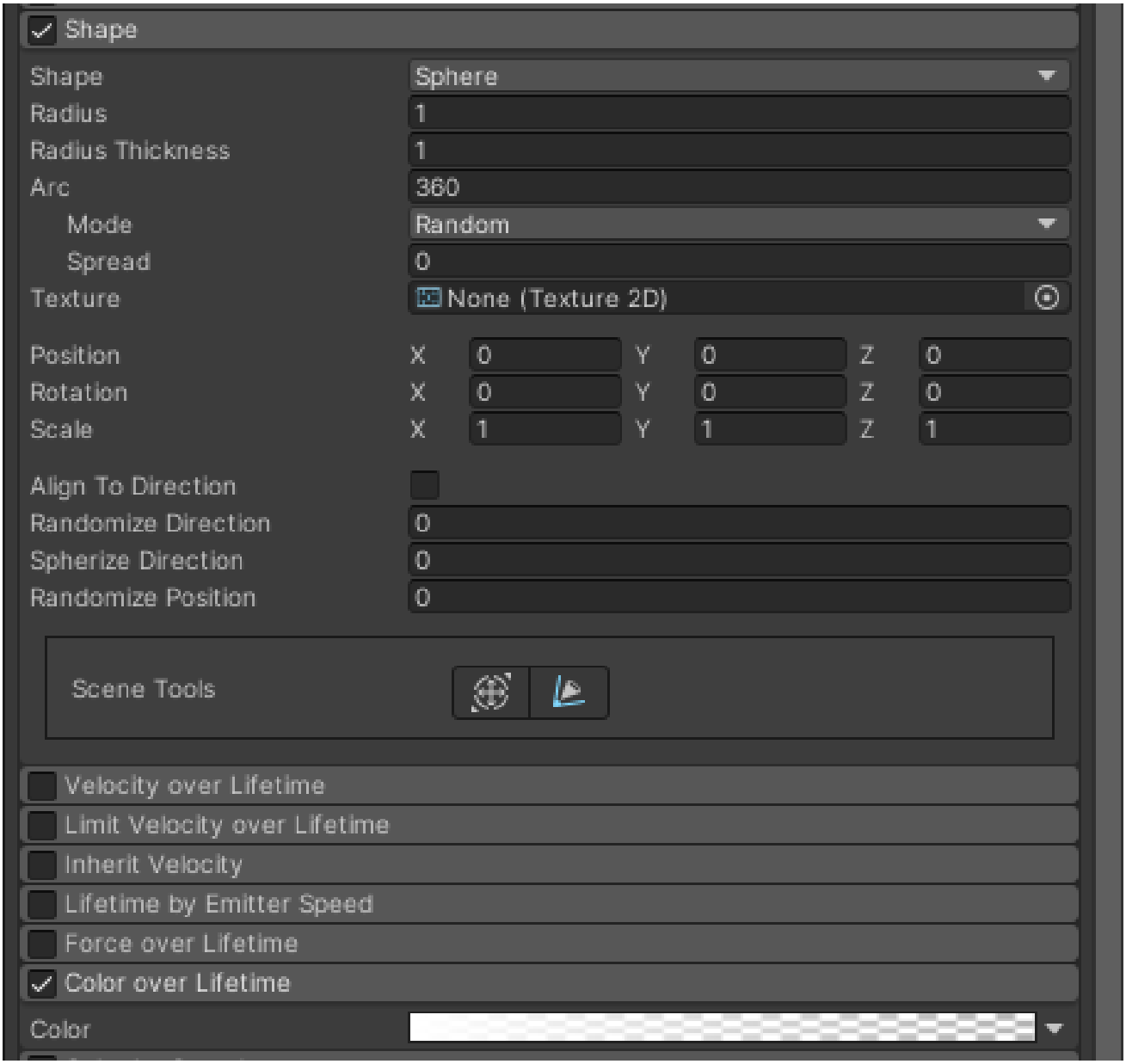


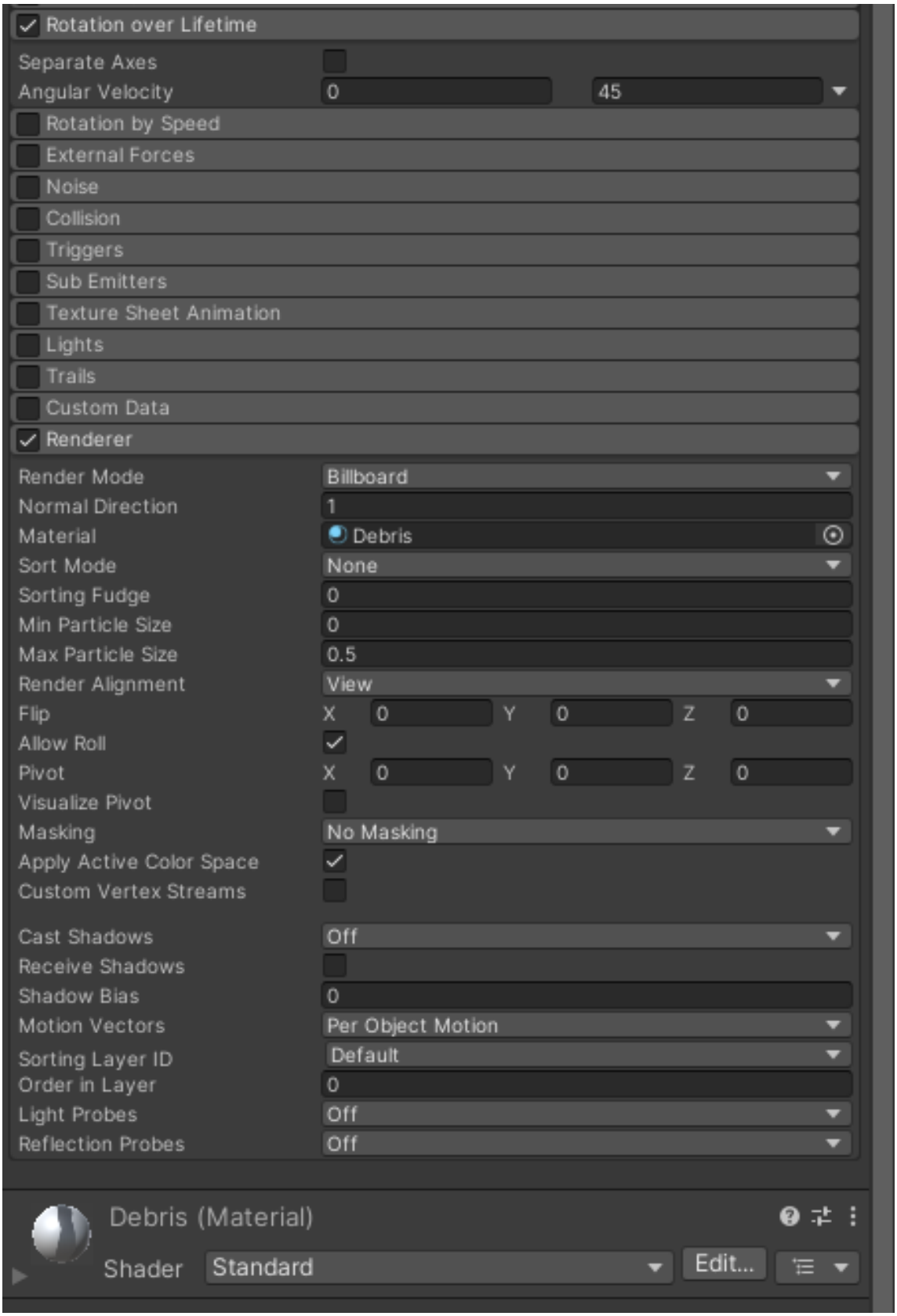
Explosion Prefab > Smoke Object Continued



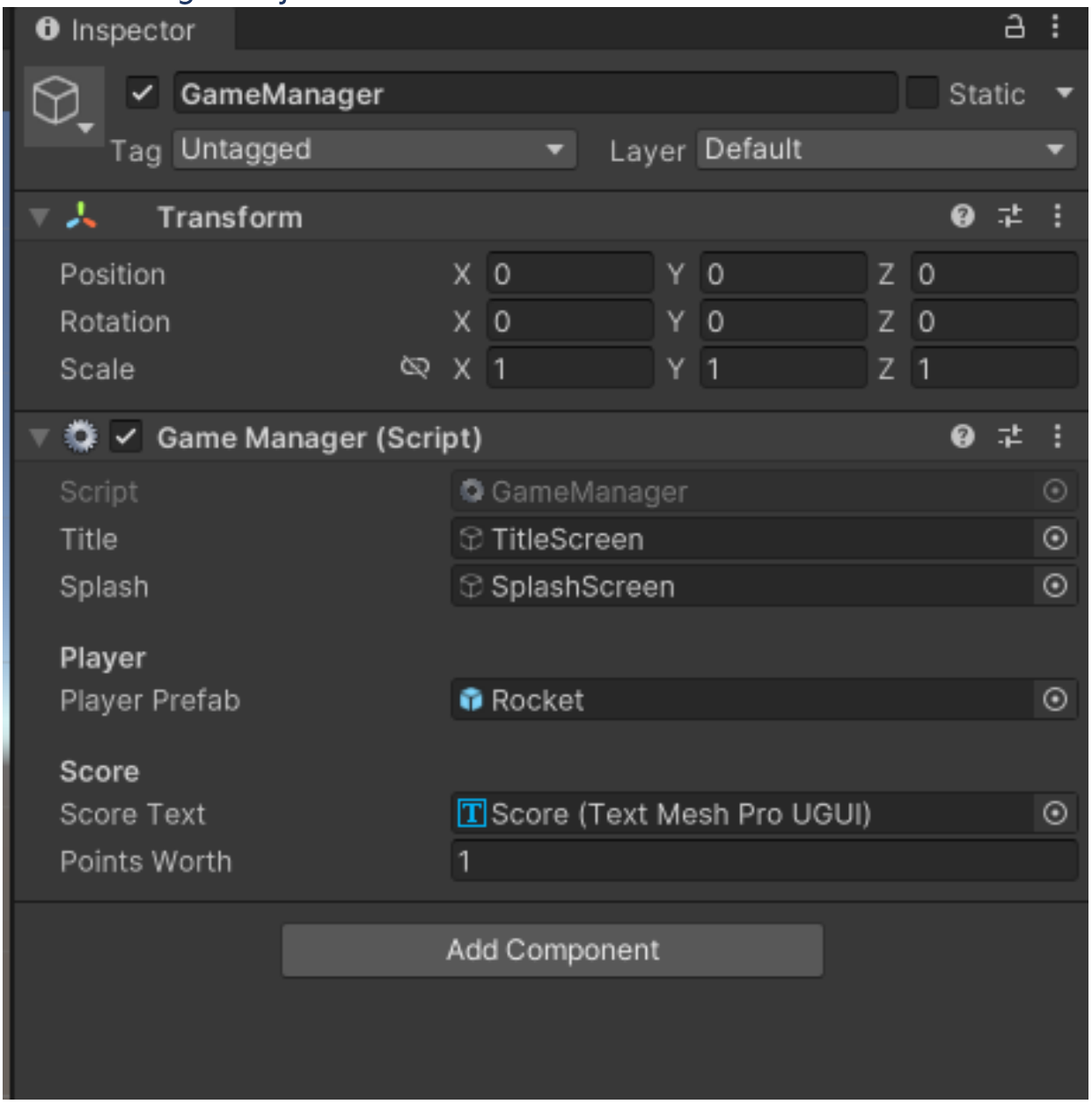
Explosion Prefab > Smoke > Debris Object







GameManager Object

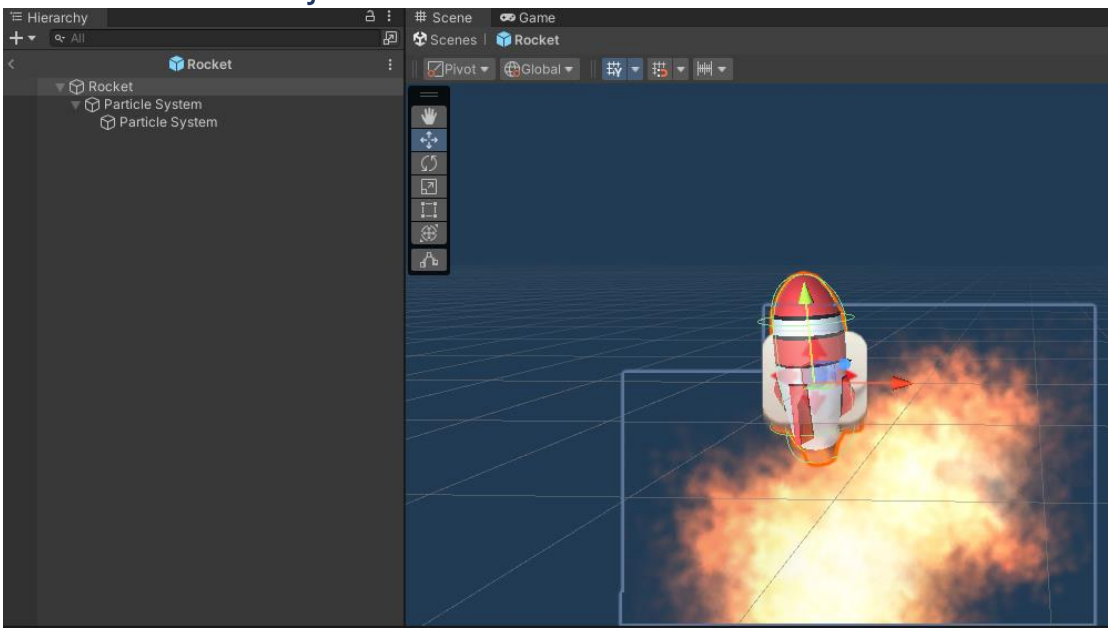


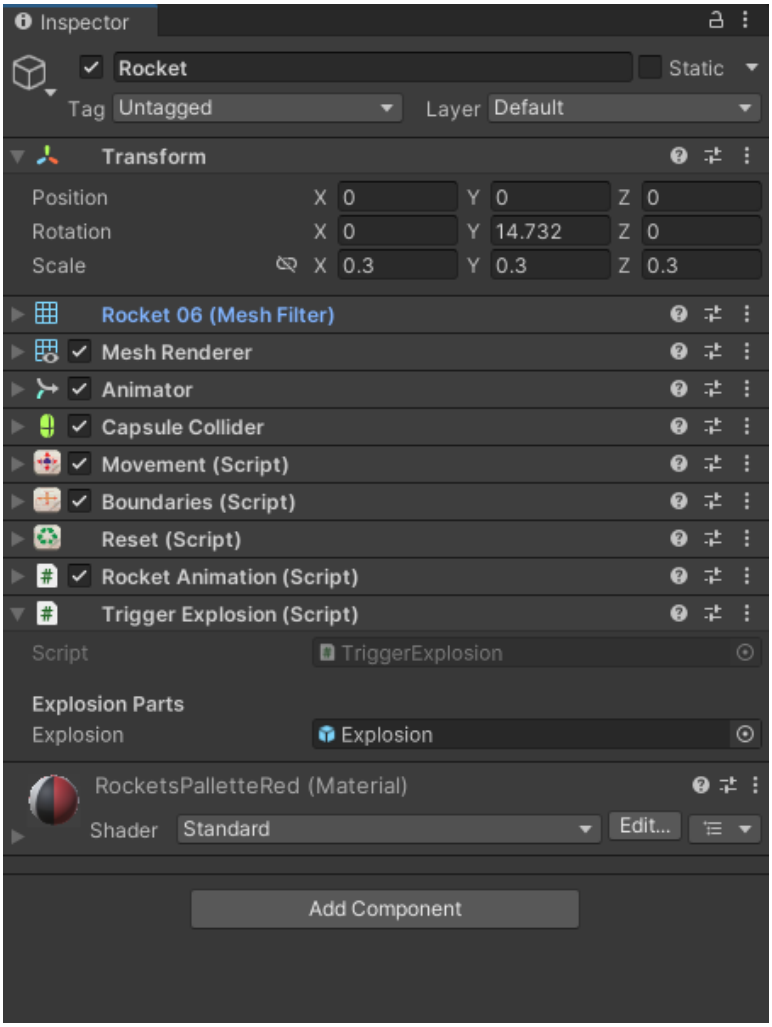
GameManager.cs Script

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

public class GameManager : MonoBehaviour
{
	private Spawner spawner;
	public GameObject title;
	private Vector2 screenBounds;

	public GameObject splash;

	[Header("Player")]
	public GameObject playerPrefab;
	private GameObject player;
	private bool gameStarted = false;

	[Header("Score")]
	public TMP_Text scoreText;
	public int pointsWorth = 1;
	private int score;

	private bool smokeCleared = true;

	private void Awake()
	{
		spawner = GameObject.Find("Spawner").GetComponent<Spawner>();
		screenBounds = Camera.main.ScreenToWorldPoint(new Vector3(Screen.width,
		Screen.height, Camera.main.transform.position.z));
		scoreText.enabled = false;
	}

	// Start is called before the first frame update
	void Start()
	{
		spawner.active = false;
		title.SetActive(true);
		splash.SetActive(false);
	}

	// Update is called once per frame
	void Update()
	{
		if (!gameStarted)
		{
			if (Input.anyKeyDown && smokeCleared)
			{
				smokeCleared = false;
				ResetGame();
			}
		}
		else
		{
			if (!player)
```

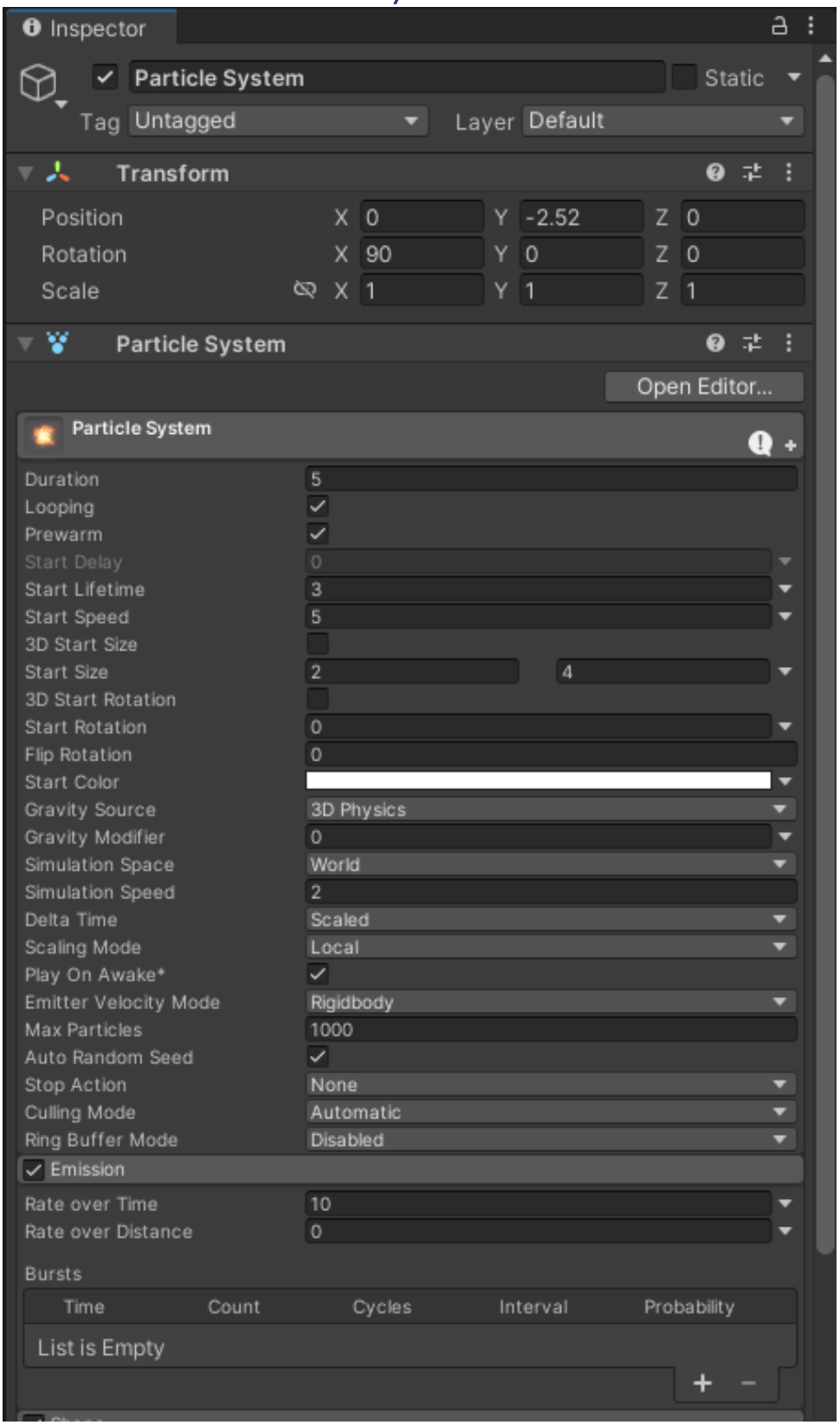
```
                {
                        OnPlayerKilled();
                }
        }

        var nextBomb = GameObject.FindGameObjectsWithTag("Bomb");

        foreach (GameObject bombObject in nextBomb)
        {
                if (bombObject.transform.position.y < (-screenBounds.y - 12))
                {
                        if (gameStarted)
                        {
                                score += pointsWorth;
                                scoreText.text = "Score: " + score.ToString();
                        }

                Destroy(bombObject);
                }
        }
    }

    void OnPlayerKilled()
    {
        spawner.active = false;
        gameStarted = false;

        splash.SetActive(true);

        Invoke("SplashScreen", 2);
    }

    void ResetGame()
    {
        spawner.active = true;
        title.SetActive(false);

        splash.SetActive(false);

        scoreText.enabled = true;
        score = 0;
        scoreText.text = "Score: " + score.ToString();
        player = Instantiate(playerPrefab,new Vector3(0,0,0),
        playerPrefab.transform.rotation);
        gameStarted = true;
    }

    void SplashScreen()
    {
        smokeCleared = true;
        splash.SetActive(true);
    }
}
```

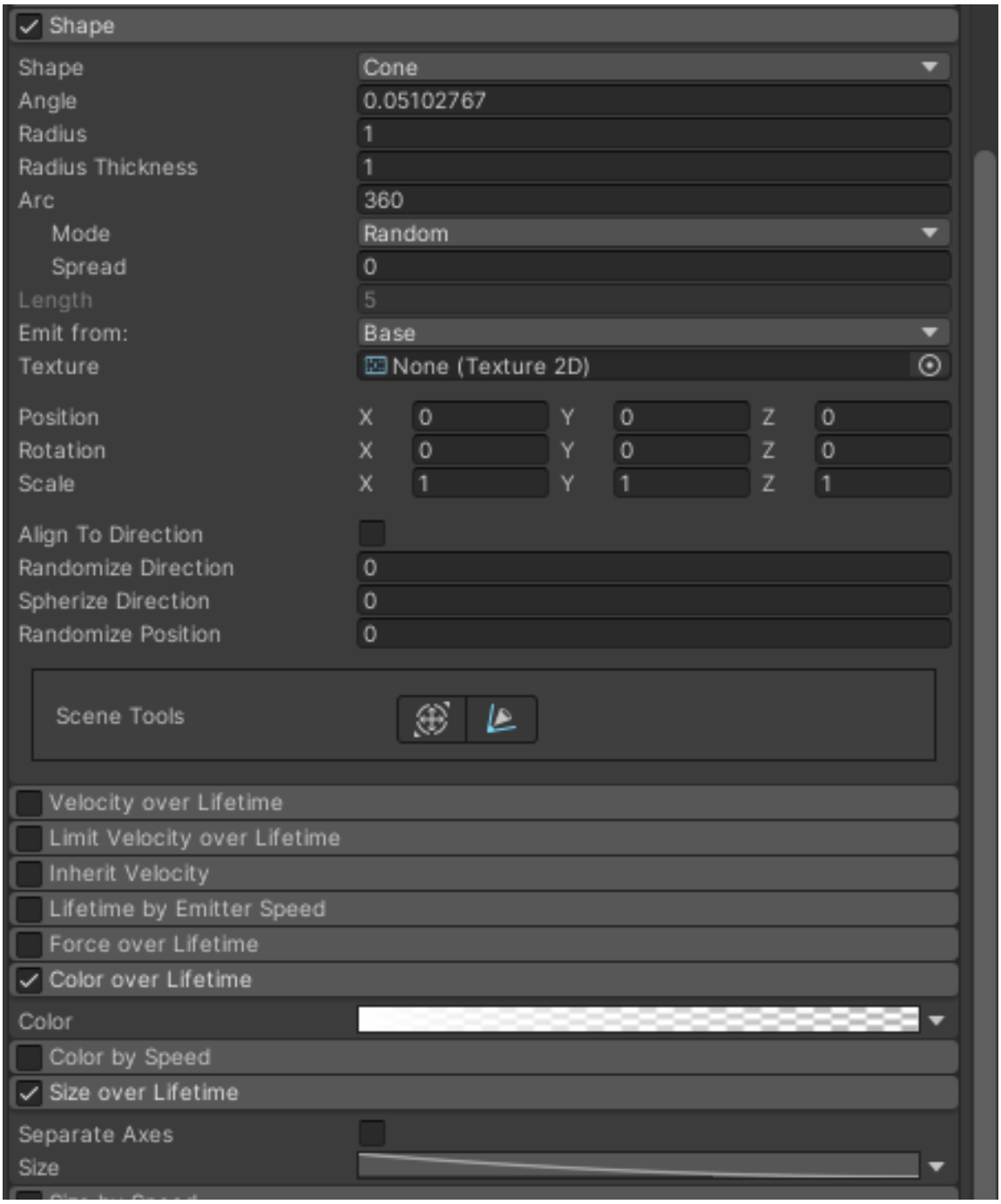
Rocket Prefab Object

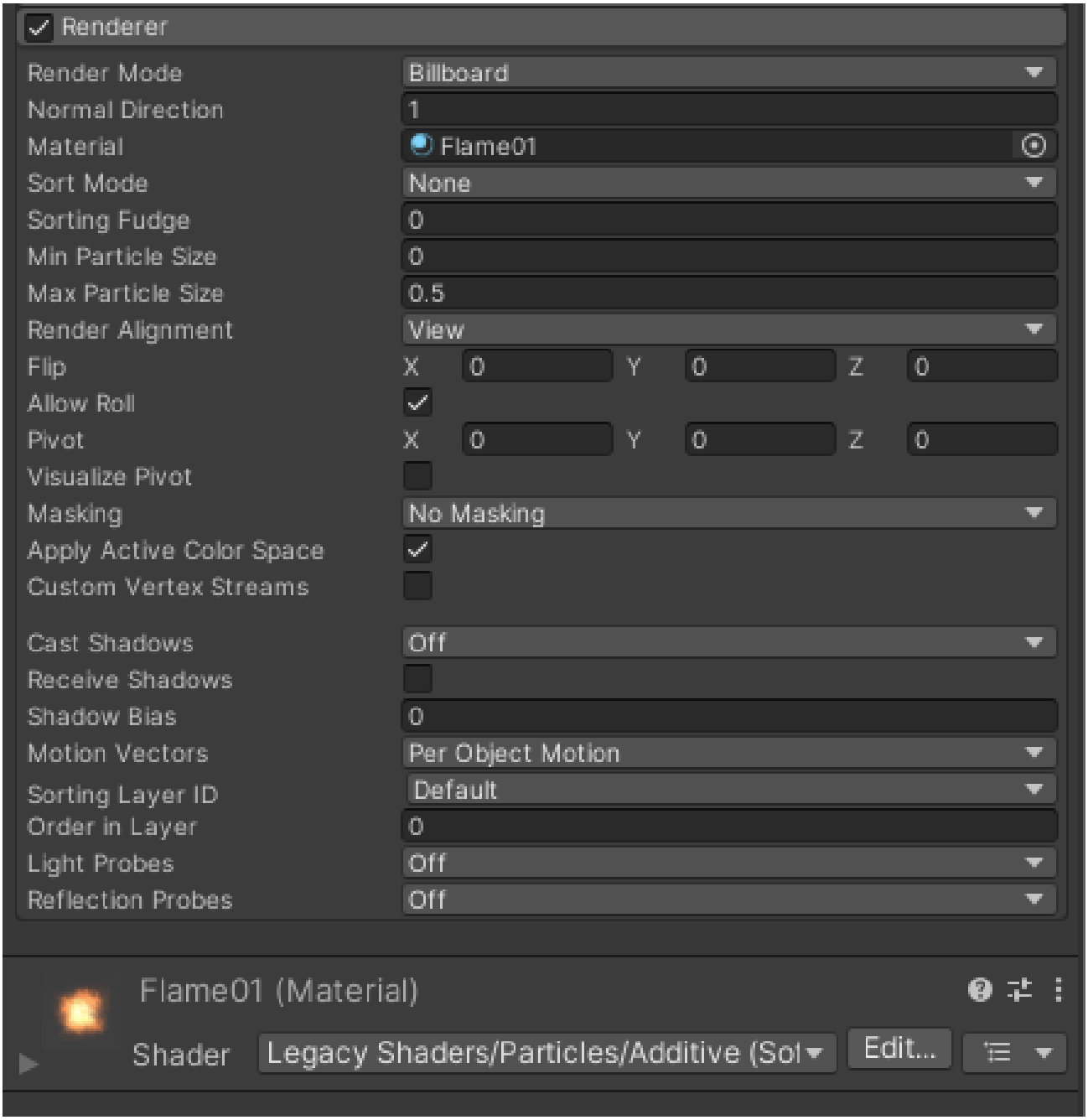




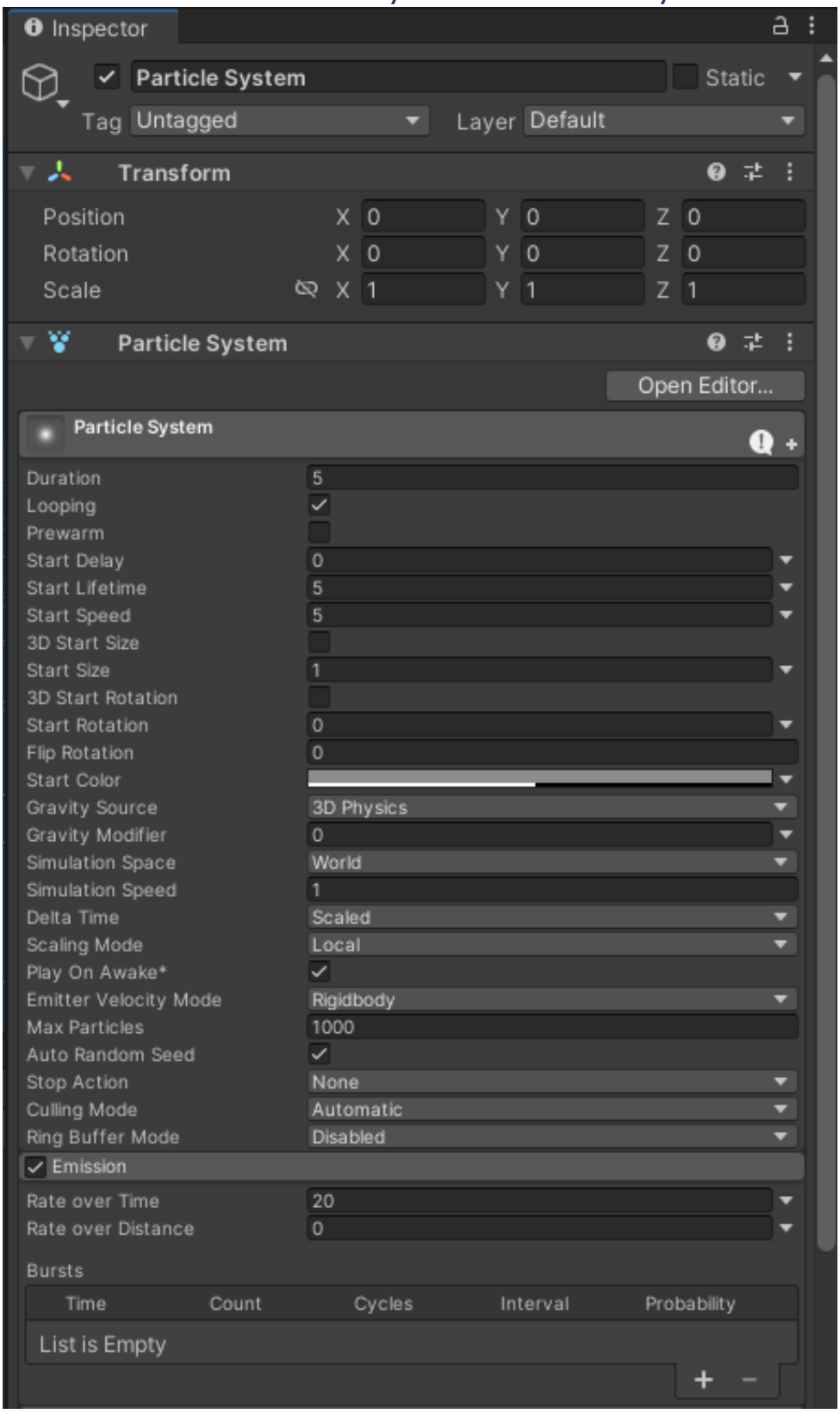
Rocket Prefab > Particle System

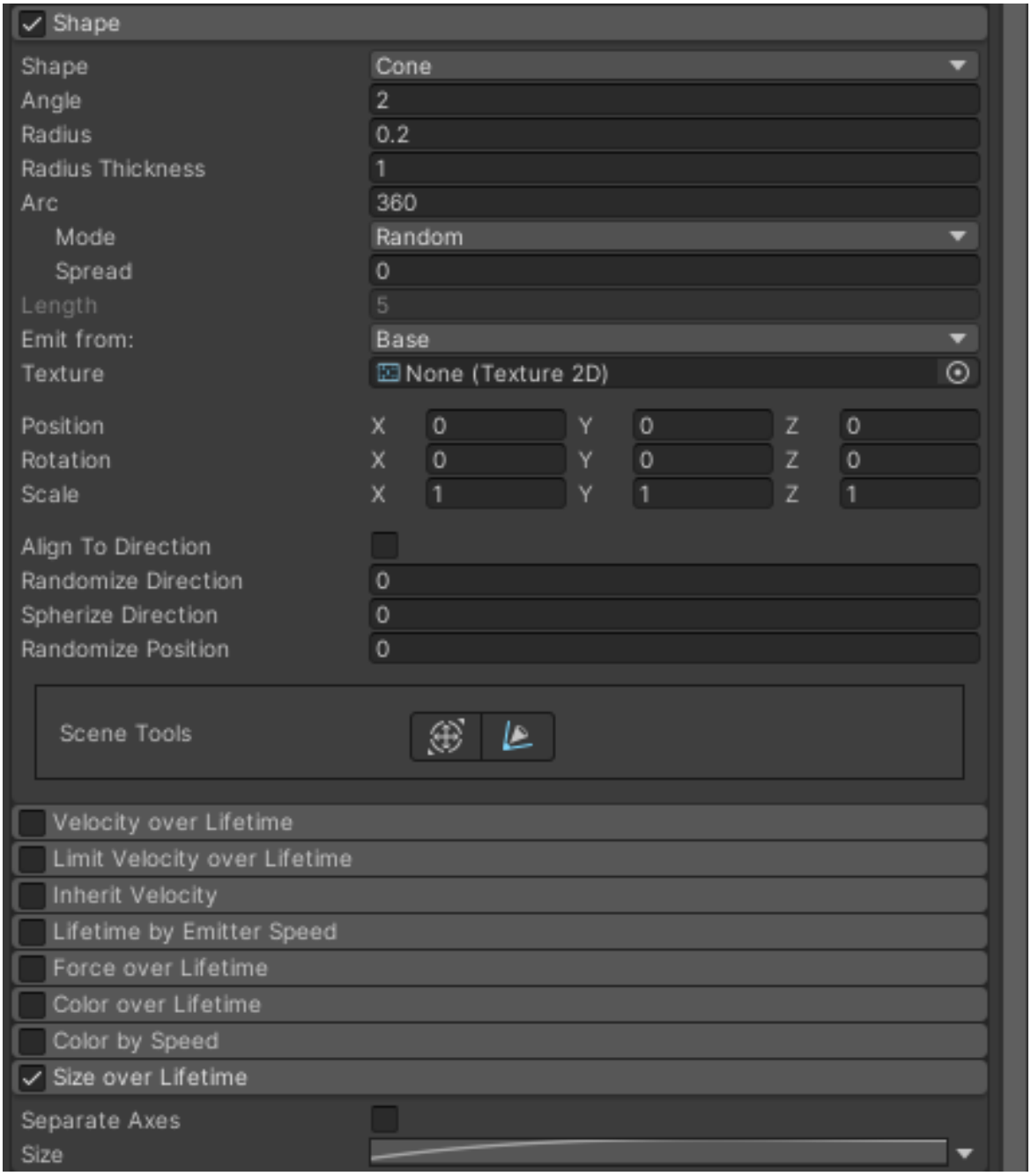


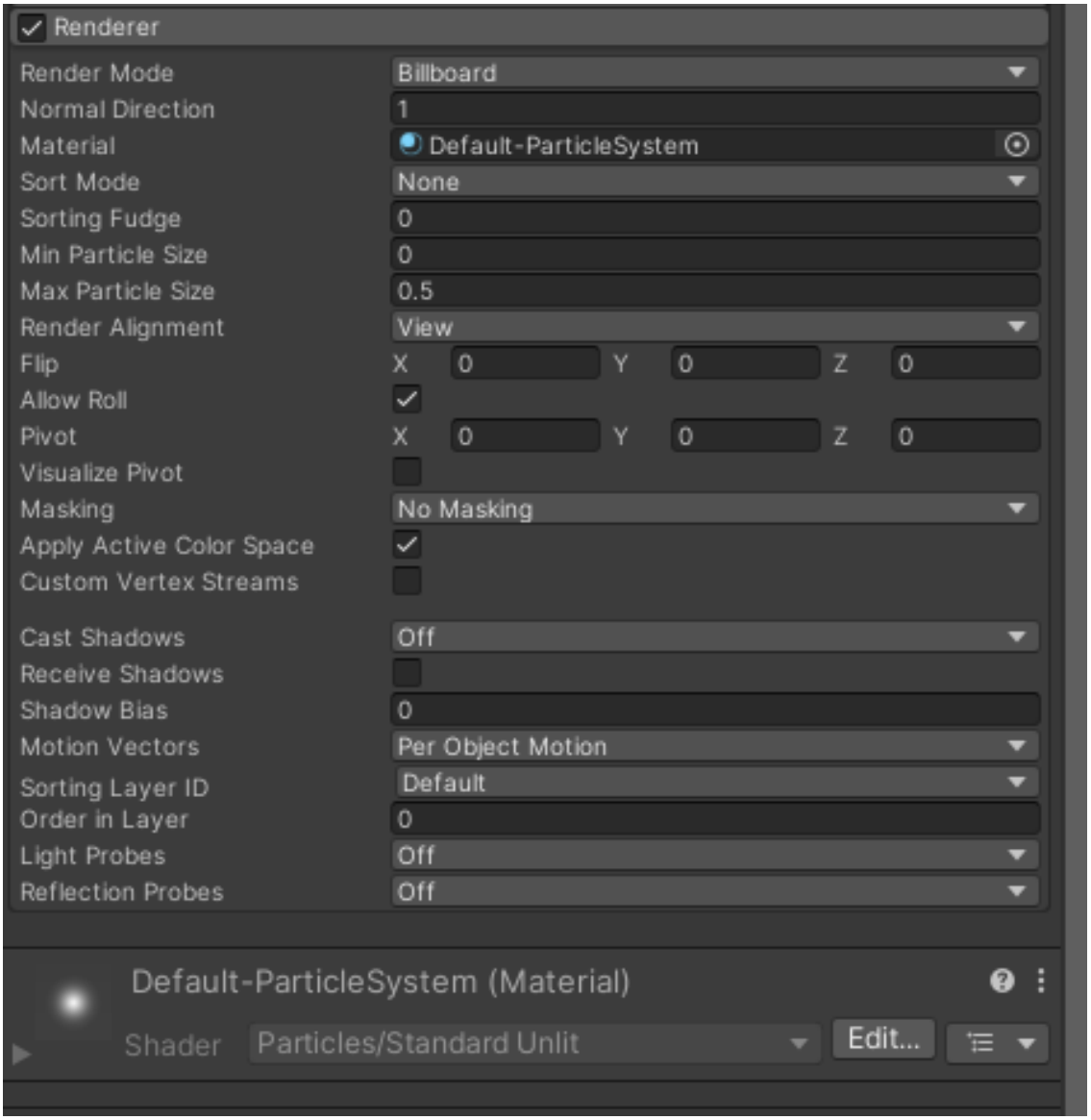




Rocket Prefab > Particle System > Particle System





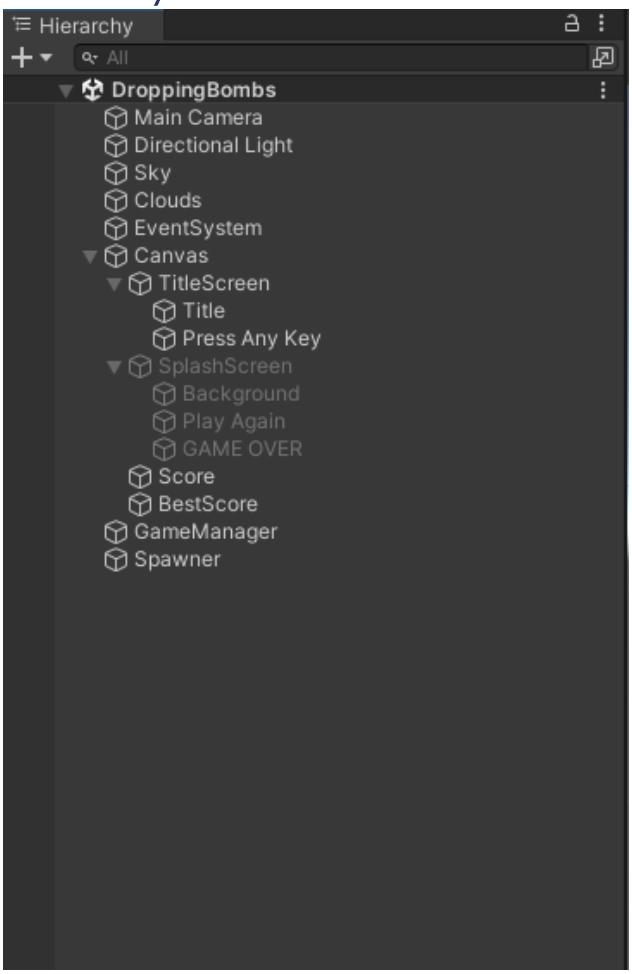


TriggerExplosion.cs Script

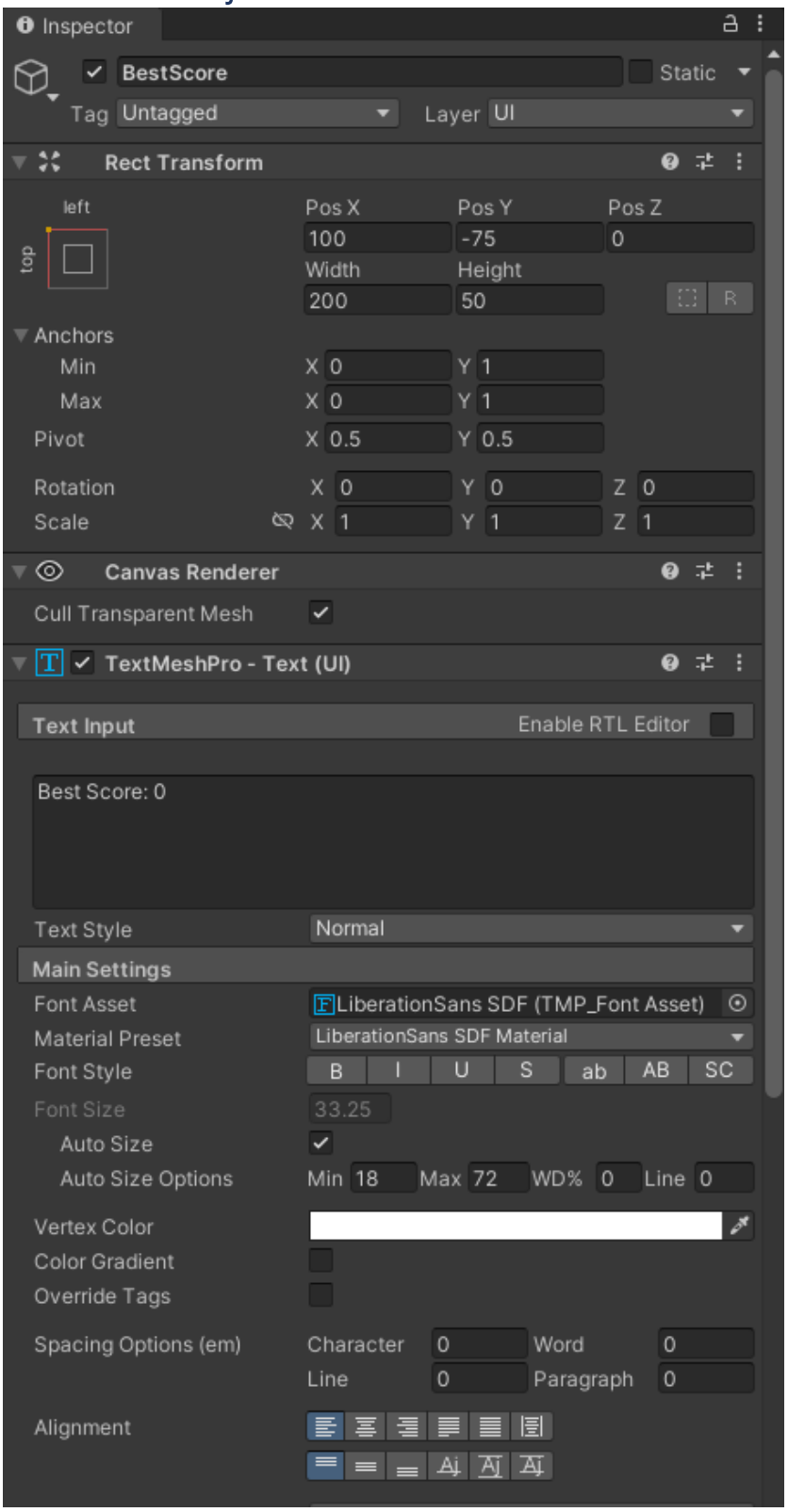
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class TriggerExplosion : MonoBehaviour

{
	[Header("Explosion Parts")]
	public GameObject explosion;

	private void OnCollisionEnter(Collision collision)
	{
		Instantiate(explosion,transform.position,transform.rotation);
	}
}
```

Activity Solution: Dropping Bombs Part 5

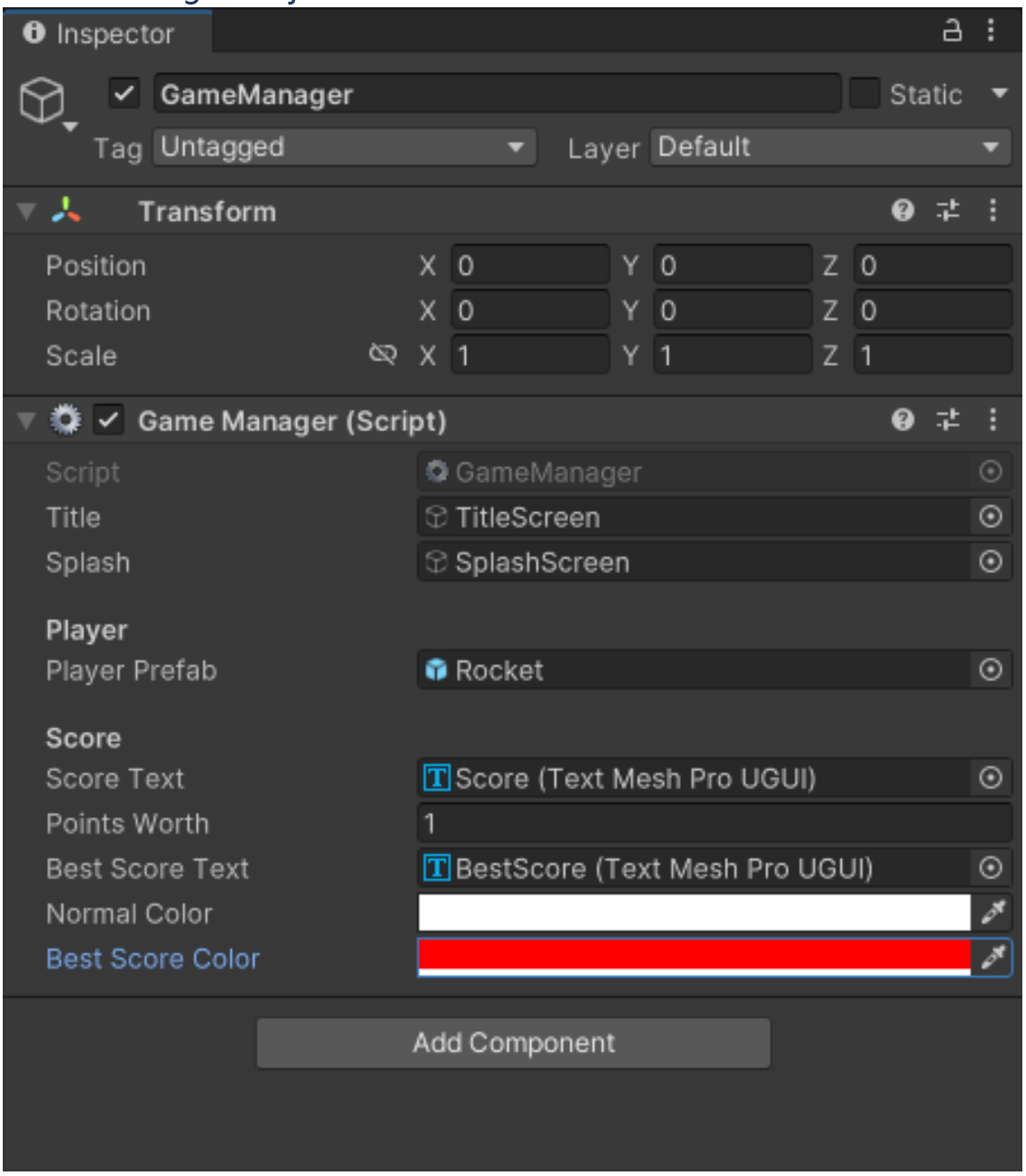
Hierarchy



BestScore Object



GameManager Object



GameManager.cs Script

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

public class GameManager : MonoBehaviour
{
	private Spawner spawner;
	public GameObject title;
	private Vector2 screenBounds;

	public GameObject splash;

	[Header("Player")]
	public GameObject playerPrefab;
	private GameObject player;
	private bool gameStarted = false;

	[Header("Score")]
	public TMP_Text scoreText;
	public int pointsWorth = 1;
	private int score;

	private int bestScore = 0;
	public TMP_Text bestScoreText;
	private bool beatBestScore;

	public Color normalColor;
	public Color bestScoreColor;

	private bool smokeCleared = true;

	private void Awake()
	{
		spawner = GameObject.Find("Spawner").GetComponent<Spawner>();
		screenBounds = Camera.main.ScreenToWorldPoint(new Vector3(Screen.width,
		Screen.height, Camera.main.transform.position.z));
		scoreText.enabled = false;

		bestScoreText.enabled = false;
	}


	// Start is called before the first frame update
	void Start()
	{
		spawner.active = false;
		title.SetActive(true);
		splash.SetActive(false);

		bestScore = PlayerPrefs.GetInt("BestScore");
		bestScoreText.text = "Best Score: " + bestScore.ToString();
	}
```

```
// Update is called once per frame
void Update()
{
	if (!gameStarted)
	{
		if (Input.anyKeyDown && smokeCleared)
		{
			smokeCleared = false;
			ResetGame();
		}
	}
	else
	{
		if (!player)
		{
			OnPlayerKilled();
		}
	}

	var nextBomb = GameObject.FindGameObjectsWithTag("Bomb");

	foreach (GameObject bombObject in nextBomb)
	{
		if (bombObject.transform.position.y < (-screenBounds.y - 12))
		{
			if (gameStarted)
			{
				score += pointsWorth;
				scoreText.text = "Score: " + score.ToString();
			}

			Destroy(bombObject);
		}
	}
}

void OnPlayerKilled()
{
	spawner.active = false;
	gameStarted = false;

	splash.SetActive(true);

	Invoke("SplashScreen", 2);

	if (score > bestScore)
	{
		bestScoreText.color = bestScoreColor;

		bestScore = score;
		PlayerPrefs.SetInt("BestScore", bestScore);
		beatBestScore = true;
		bestScoreText.text = "Best Score: " + bestScore.ToString();
	}
}

void ResetGame()
{
```

```
            bestScoreText.color = normalColor;

            spawner.active = true;
            title.SetActive(false);

            splash.SetActive(false);

            scoreText.enabled = true;
            score = 0;
            scoreText.text = "Score: " + score.ToString();

            beatBestScore = false;
            bestScoreText.enabled = true;

            player = Instantiate(playerPrefab,new Vector3(0,0,0),
            playerPrefab.transform.rotation);
            gameStarted = true;
        }

        void SplashScreen()
        {
            smokeCleared = true;
            splash.SetActive(true);
        }
}
```