Lab 7 – Asteroids – Adding a GUI System
Jay Popat 22346566

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

public class Asteroid : MonoBehaviour {

    // inspector settings
    public Rigidbody rigidBody;
    void Start () {
        // randomise velocity
        rigidBody.velocity = new Vector3 (Random.Range (-10f, 10f), 0f,
Random.Range (-10f, 10f));
        rigidBody.angularVelocity = new Vector3 (Random.Range (-4f, 4f),
Random.Range (-4f, 4f), Random.Range (-4f, 4f));
    }

    public void SetScale(float min, float max) {
        transform.localScale = new Vector3(Random.Range(min,max),
Random.Range(min,max), Random.Range(min,max));
        rigidBody.mass = transform.localScale.x * transform.localScale.y *
transform.localScale.z;
    }

    void OnCollisionEnter(Collision collision) {
        if (!collision.gameObject.name.Contains("asteroid")) {
            Spaceship ss = collision.gameObject.GetComponent<Spaceship> ();
            if (ss != null && ss.isInvulnerable)
                return;

            GameManager.instance.AddScore(10);

            // we've collided with something other than another asteroid
            Destroy(collision.gameObject); // if it's the player spaceship, the
Spaceship scripts' OnDestroy will look after re-creating it
            Destroy(this.gameObject);

            if (rigidBody.mass > 0.00015f) {
                float minScale = rigidBody.mass * 50f;
                float maxScale = minScale * 2f;
                for (int i = 0; i < 3; i++) {
                    GameObject go = Instantiate
(GameManager.instance.asteroidPrefab) as GameObject;
                    go.transform.position = transform.position;
                    go.GetComponent<Asteroid> ().SetScale (minScale, maxScale);
                }
            }
        }
    }
}
using System.Collections;
using UnityEngine;
```

```
public class AutoDestroy : MonoBehaviour
{
    public float minLifetime, maxLifetime;

    private void Start()
    {
        StartCoroutine(HandleLifetime());
    }

    private IEnumerator HandleLifetime()
    {
        yield return new WaitForSeconds(Random.Range(minLifetime, maxLifetime));
        Destroy(gameObject);
    }
}
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Bullet : MonoBehaviour {

    // inspector settings
    public Rigidbody rigid;
    //

    // Use this for initialization
    void Start () {
        rigid.velocity = transform.forward * 30f;
    }

    // Update is called once per frame
    void Update () {

    }
}
using UnityEngine;

public enum GameState
{
    Menu,
    Playing
}
public class GameManager : MonoBehaviour
{
//
// class-level statics
    public static GameManager instance;
    public static int currentGameLevel;
    public static Vector3 screenBottomLeft, screenTopRight;
    public int score;
    public int highScore;
    public int lives;
    public GameState gameState;

    public static float screenWidth, screenHeight;
```

```
// inspector settings
    public GameObject asteroidPrefab, spaceshipPrefab, fragmentPrefab,
bulletPrefab;

//
// Use this for initialization
    private void Start()
    {
        instance = this;
        gameState = GameState.Menu;
        highScore = PlayerPrefs.GetInt("highScore", 0);

    }
    public void StartNewGame()
    {
        Camera.main.transform.position = new Vector3(0f, 30f, 0f);
        Camera.main.transform.LookAt(Vector3.zero, new Vector3(0f, 0f, 1f));
        currentGameLevel = 0;
        score = 0;
        lives = 3;
        StartNextLevel();
        CreatePlayerSpaceship();
        gameState = GameState.Playing;
        GameObject.FindObjectOfType<GameUI>().SetPlaying(false);
    }

    private void endGame(){
        gameState = GameState.Menu;
        if (score > highScore)
        {
            highScore = score;
            PlayerPrefs.SetInt("highScore", highScore);
        }
        score = 0;
        lives = 3;
        currentGameLevel = 0;
        StartNextLevel();
        CreatePlayerSpaceship();

        foreach (var asteroid in GameObject.FindGameObjectsWithTag("Asteroid"))
        {
            Destroy(asteroid);
        }
        GameObject.FindObjectOfType<GameUI>().SetPlaying(false);
    }
    public void AddScore(int points)
    {
        score += points;
        if (score > highScore)
        {
            highScore = score;
        }

    }
```

```
    public void LoseLife()
    {
        lives--;
        if (lives <= 0)
        {
            endGame();
        }
        else
        {
            CreatePlayerSpaceship();
        }
    }
    public static void StartNextLevel()
    {
        currentGameLevel++;
        screenBottomLeft = Camera.main.ViewportToWorldPoint(new Vector3(-0.1f, -
0.1f, 30f));
        screenTopRight = Camera.main.ViewportToWorldPoint(new Vector3(1.1f,
1.1f, 30f));
        screenWidth = screenTopRight.x - screenBottomLeft.x;
        screenHeight = screenTopRight.z - screenBottomLeft.z;
        Debug.Log("BottomLeft: " + screenBottomLeft);
        Debug.Log("TopRight: " + screenTopRight);
        Debug.Log("Width: " + screenWidth);
        Debug.Log("Height: " + screenHeight);
// instantiate some asteroids near the edges of the screen
        for (var i = 0; i < currentGameLevel * 2 + 3; i++)
        {
            var go = Instantiate(instance.asteroidPrefab);
            float x, z;
            if (Random.Range(0f, 1f) < 0.5f)
                x = screenBottomLeft.x + Random.Range(0f, 0.15f) * screenWidth;
// near the left edge
            else
                x = screenTopRight.x - Random.Range(0f, 0.15f) * screenWidth; //
near the right edge
            if (Random.Range(0f, 1f) < 0.5f)
                z = screenBottomLeft.z + Random.Range(0f, 0.15f) * screenHeight;
// near the bottom edge
            else
                z = screenTopRight.z - Random.Range(0f, 0.15f) * screenHeight;
// near the top edge
            go.transform.position = new Vector3(x, 0f, z);
        }
    }

    public static void CreatePlayerSpaceship()
    {
// instantiate the player's spaceship
        GameObject go = Instantiate(instance.spaceshipPrefab) as GameObject;
        go.transform.position = Vector3.zero;
    }
}
using UnityEngine;
```

```
public class GameUI : MonoBehaviour
{

    private bool _isPlaying = false;

    private void OnGUI()
    {
        if (!_isPlaying)
        {
            // Menu GUI
            GUILayout.BeginArea(new Rect(Screen.width / 2 - 50, Screen.height /
2 - 25, 100, 50));
            if (GUILayout.Button("Play"))
            {
                GameManager.instance.StartNewGame();
                _isPlaying = true;
            }
            GUILayout.EndArea();
        }
        else
        {
            // Playing GUI
            GUILayout.BeginArea(new Rect(10, 10, 200, 100));
            GUILayout.Label("Score: " + GameManager.instance.score);
            GUILayout.Label("High Score: " + GameManager.instance.highScore);
            GUILayout.Label("Lives: " + GameManager.instance.lives);
            GUILayout.EndArea();
        }
    }

    public void SetPlaying(bool playing)
    {
        _isPlaying = playing;
    }
}
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Spaceship : MonoBehaviour {

    // inspector settings
    public Rigidbody rigidBody;
    public GameObject bulletPrefab;
    //

    // public member data
    [HideInInspector] public bool isInvulnerable = true;
    //

    // private member data
    private float lastFiredTime = 0f;
    //

    void Start() {
```

```
        Invoke ("MakeVulnerable", 2f);
    }

    private void MakeVulnerable() {
        isInvulnerable = false;
    }

    void FixedUpdate () {
        if (Input.GetKey(KeyCode.UpArrow))
            rigidBody.AddForce(transform.forward * (rigidBody.mass *
Time.fixedDeltaTime * 800f));

        if (Input.GetKey(KeyCode.LeftArrow))
            rigidBody.AddTorque(-transform.up * (rigidBody.mass * Time.deltaTime
* 1500f));
        else if (Input.GetKey(KeyCode.RightArrow))
            rigidBody.AddTorque(transform.up * (rigidBody.mass * Time.deltaTime
* 1500f));

        // firing is only allowed at most once per 0.25 seconds
        if (Input.GetKey (KeyCode.Space) && lastFiredTime + 0.25f ⩽ Time.time)
{
            lastFiredTime = Time.time;
            FireBullet ();
        }
    }

    void OnDestroy() {
        GameManager.instance.LoseLife();
        GameManager.CreatePlayerSpaceship();
    }

    private void FireBullet() {
        GameObject go = Instantiate (bulletPrefab) as GameObject;
        go.transform.position = transform.position + transform.forward*3f;
        go.transform.rotation = transform.rotation;
    }
}
```