```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using UnityEngine.SceneManagement;
5
6  public class HorizontalScript : MonoBehaviour
7  {
8      // Get reference variables
9      Transform playerTrans;   // Player's Transform component
10     GameController gc;       // Instance of Game Controller script
11     Renderer attackRend;     // The attack's renderer
12     Transform attackTrans;   // The attack's Transform
13
14     // Lifetime of the attack
15     public float lifetime;
16
17     // Warning phase variables
18     bool warningPhase;            // Is it the warning phase
19     public float warningLength;   // Length of the warning phase
20     float timeAcc;                // Time accumulation (used in the fade in)
21     public float startLerp;       // Start value of the fade lerp
22     public float endLerp;         // End value of the fade lerp
23     public Color startColor;      // Start color of fade lerp
24     public Color endColor;        // End color of fade lerp
25
26     // Move phase variables
27     bool movePhase;           // Is it move phase
28     bool moveRight;            // Is the attack moving from left to right
29     public float moveSpeed;   // Speed of the attack
30
31
32     // Gets the direction the attack will move in (true = left to right, false
          = right to left)
33     bool GetDirection() {
34
35         // Get distance of player to left and right sides of screen
36         float playerDistToLeftSide = Vector3.Distance(playerTrans.position,
             new Vector3(-18, playerTrans.position.y, 0));
37         float playerDistToRightSide = Vector3.Distance(playerTrans.position,
             new Vector3(18, playerTrans.position.y, 0));
38
39         // Test if player is closer to right or left side of screen
40         if (playerDistToRightSide <= playerDistToLeftSide) {
41             // True means go left to right
42             return true;
43         } else {
44             // False means go right to left
45             return false;
46         }
```

```
47      }
48
49      // Start is called before the first frame update
50      void Start()
51      {
52          // Get components for variables
53          playerTrans = GameObject.FindGameObjectWithTag("Player").transform;
54          attackRend = GetComponent<Renderer>();
55          attackTrans = GetComponent<Transform>();
56          gc = GameObject.FindGameObjectWithTag
              ("GameController").GetComponent<GameController>();
57
58          // Set up warning phase
59          warningPhase = true;
60          timeAcc = startLerp;
61
62          // Set up move phase
63          movePhase = false;
64          moveRight = GetDirection();
65      }
66
67      // Update is called once per frame
68      void Update()
69      {
70          // If currently warning phase
71          if (warningPhase) {
72
73              // If warning phase has ended
74              if (timeAcc >= endLerp) {
75
76                  // Set color to endLerp in case the timeAcc goes over
77                  attackRend.material.color = Color.Lerp(startColor, endColor,
                      endLerp);
78
79                  // End warning phase
80                  warningPhase = false;
81
82                  // Set up move phase
83                  if (moveRight) {
84                      attackTrans.position = new Vector3(17,
                        attackTrans.position.y, 0);
85                  } else {
86                      attackTrans.position = new Vector3(-17,
                        attackTrans.position.y, 0);
87                  }
88
89                  // Set color to full red and start move phase
90                  attackRend.material.color = endColor;
91                  movePhase = true;
```

```
92
93
94              } else {
95
96                  // Set color for warning
97                  attackRend.material.color = Color.Lerp(startColor, endColor,
                      timeAcc);
98
99                  // Time.deltaTime is the time since the last frame. In a
                      perfect world, if the game was running
100                 // at 60 frames per second then after 1 second (ingnoring the
                      division) timeAcc would equal 60
101                 // However, since fade-in ends when timeAcc >= endLerp (0.6),
                      we'll never be able to change the
102                 // fade-in length from 0.6 seconds. To scale Time.deltaTime,
                      it's divided by warningLength.
103                 timeAcc += Time.deltaTime / warningLength;
104             }
105         }
106
107
108         // If currently move phase
109         if (movePhase) {
110
111             // If moving left to right...
112             if (moveRight) {
113
114                 // Go in that direction
115                 attackTrans.Translate(Vector3.left * (Time.deltaTime *
                      moveSpeed));
116
117                 // If attack has reached the middle of the screen
118                 if (attackTrans.position.x <= 0) {
119
120                     // End the move phase
121                     movePhase = false;
122                     // Set the position to the center of the screen beacuse if
                          the speed is really fast it can
123                     // go past the middle
124                     attackTrans.position = new Vector3(0,
                        attackTrans.position.y, attackTrans.position.z);
125
126                     // Waits for the length of lifetime, then destroys itself
127                     StartCoroutine(Suicide());
128                 }
129
130             // If moving right to left...
131             } else {
132
```

```
133                 // Move in the other direction
134                 attackTrans.Translate(Vector3.right * (Time.deltaTime *
                      moveSpeed));
135
136                 // If attack has reached the middle of the screen
137                 if (attackTrans.position.x >= 0) {
138
139                     // End the move phase
140                     movePhase = false;
141                     // Set the position to the cente of the screen because if
                    the speed is really fast it can
142                     // go past the middle
143                     attackTrans.position = new Vector3(0,
                    attackTrans.position.y, attackTrans.position.z);
144
145                     // Waits for the length of lifetime, then destroys itself
146                     StartCoroutine(Suicide());
147                 }
148             }
149         }
150     }
151
152     IEnumerator Suicide() {
153         // Wait for the length of lifetime
154         yield return new WaitForSeconds(lifetime);
155
156         // Destoy this game object
157         Destroy(gameObject);
158     }
159
160     // If something enters the attacks collider...
161     private void OnTriggerEnter(Collider other) {
162
163         // Check if collided object has tag "Player"
164         if (other.tag == "Player") {
165
166             // End the game
167             gc.EndGame();
168         }
169     }
170 }
171
```