

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class PlayerAttack : PlayerAnimation
6 {
7     //itens
8     public GameObject[] weapon;
9
10    //particles
11    [Header ("Particulas")]
12    public GameObject[] particle;
13
14
15    [Header("SkillsPrefabs")]
16    public GameObject powerPrefab;
17    public GameObject earthShatterPrefab;
18
19    [HideInInspector]
20    public bool canAttack;
21
22    private bool nextAttack = true;
23    private bool rangeAttack = true;
24
25    public AvatarMask[] attackMask;
26
27    private void Awake()
28    {
29        canAttack = true;
30    }
31
32    void Update ()
33    {
34        Debug.Log(canAttack);
35
36        if (canAttack)
37        {
38            InputAttacks();
39            ParticleSystem();
40            ActivateWeapon();
41        }
42    }
43
44
45
46    private void InputAttacks()
47    {
48        Attack(1, Input.GetButton("Fire1") && nextAttack);
49        Attack(2, Input.GetButton("Fire2") && nextAttack);
50
51        Skill(1, Input.GetKey(KeyCode.Q) && nextAttack);
52        Skill(2, Input.GetKey(KeyCode.E) && nextAttack);
53        Skill(3, Input.GetKey(KeyCode.R) && nextAttack);
54
55        if (Input.GetButton("Fire1") && nextAttack)
56        {
```

```
57
58         StartCoroutine(OnCompleteAttackAnimation(0.5f));
59     }
60     else if(Input.GetButton("Fire2") && nextAttack)
61     {
62         StartCoroutine(OnCompleteAttackAnimation(0.5f));
63     }
64     else if (Input.GetKey(KeyCode.Q) && nextAttack)
65     {
66         rangeAttack = true;
67         StartCoroutine(OnCompleteAttackAnimation(0.7f));
68     }
69     else if (Input.GetKey(KeyCode.E) && nextAttack)
70     {
71         rangeAttack = true;
72         StartCoroutine(OnCompleteAttackAnimation(0.8f));
73     }
74     else if (Input.GetKey(KeyCode.R) && nextAttack)
75     {
76         StartCoroutine(OnCompleteAttackAnimation(0.5f));
77     }
78
79 }
80
81 private void ParticleSystem()
82 {
83     particle[0].SetActive(isAttacking(1));
84     particle[1].SetActive(isAttacking(2));
85     particle[2].SetActive(isAttacking(5) && rangeAttack);
86     particle[4].SetActive(isAttacking(7));
87
88 }
89 private void RangeAttack()
90 {
91     //Create the power from powerprefab
92     var power = (GameObject)Instantiate(
93         powerPrefab,
94         particle[2].transform.position,
95         particle[2].transform.rotation);
96
97     //add velocity to the power
98     power.GetComponent<Rigidbody>().velocity = this.transform.forward * 10;
99
100     //destroy the bullet after
101     Destroy(power, 1f);
102 }
103 private void MeleeRangeAttack()
104 {
105     //Create the power from powerprefab
106     var power = (GameObject)Instantiate(
107         earthShatterPrefab,
108         particle[3].transform.position,
109         particle[3].transform.rotation);
110
111     //add velocity to the power
```

```
112         //power.GetComponent<Rigidbody>().velocity = this.transform.forward * 10;
113
114         //destroy the bullet after
115         Destroy(power, 1.5f);
116     }
117
118     private void ActivateWeapon()
119     {
120         weapon[0].SetActive(!isAttacking(0) && !isAttacking(5));
121         weapon[1].SetActive(isAttacking(0) || isAttacking(5));
122     }
123
124     IEnumerator Rotate(float duration, bool isRotate)
125     {
126         if(isRotate)
127         {
128             Quaternion startRot = transform.rotation;
129             float t = 0.0f;
130             while (t < duration)
131             {
132                 t += Time.deltaTime;
133                 transform.rotation = startRot * Quaternion.AngleAxis(t /
134                                     duration * 360f, transform.up); //or transform.right if you
135                                     want it to be locally based
136                 yield return null;
137             }
138             transform.rotation = startRot;
139         }
140     }
141
142     IEnumerator OnCompleteAttackAnimation(float coldown)
143     {
144         nextAttack = false;
145         yield return new WaitForSeconds(coldown);
146         if(isAttacking(5) && rangeAttack)
147         {
148             RangeAttack();
149             rangeAttack = false;
150         }
151         if (isAttacking(6) && rangeAttack)
152         {
153             MeleeRangeAttack();
154             rangeAttack = false;
155         }
156         nextAttack = true;
157     }
158 }
```