

State.cs

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  // State 추상클래스
6  public abstract class State : MonoBehaviour
7  {
8      public abstract void DoAction(MyState state);
9  };
10
11  // 상태 인터페이스 클래스
12  //public interface State
13  //{
14  //    void DoAction();
15  //};
```

ConcreteStateStand.cs

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  // ConcreteState 클래스 : 서기
6  public class ConcreteStateStand : State {
7
8      public override void DoAction(MyState state)
9      {
10         Debug.Log("Stand !!!");
11         StartCoroutine(HandleStand(state));
12     }
13
14     IEnumerator HandleStand(MyState state)
15     {
16         transform.eulerAngles = new Vector3(0, 90, 0);
17         transform.position = new Vector3(0, 1.0f, 0);
18         yield return null;
19     }
20 }
```

ConcreteStateJump.cs

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  // ConcreteState 클래스 : 점프
6  public class ConcreteStateJump : State
7  {
8      float gravity = 0.0f;    // 중력의 값
9      Vector3 velocity;        // 캐릭터의 현재 높이 저장값
10     const int MAX_CHARGE = 20;
11 }
```

```

12     public override void DoAction(MyState state)
13     {
14         Debug.Log("Jump !!!");
15         velocity = transform.position;
16         StartCoroutine(HandleJump(state));
17     }
18
19     IEnumerator HandleJump(MyState state)
20     {
21         gravity = 0.7f;
22
23         while (true)
24         {
25             if (state == MyState.STATE_DIVING)
26             {
27                 break;
28             }
29
30             velocity.y = velocity.y + gravity;
31
32             transform.position = velocity;
33
34             if (velocity.y < 1.0f)
35             {
36                 break;
37             }
38
39             gravity = gravity - 0.1f;
40
41             yield return new WaitForSeconds(0.05f);
42         }
43
44         gravity = 0.0f;
45         velocity.y = 1.0f;
46         transform.position = velocity;
47         GetComponent<MyAction8>().setActionType(MyState.STATE_STANDING);
48         // 위 코드 대신 점프 후 서 있는 상태 하나를 더 만들 수도 있다.
49         //state = STATE.STATE_STANDING2;
50
51         yield return null;
52     }
53 }

```

ConcreteStateDown.cs

```

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  // ConcreteState 클래스 : 앞드리기
6  public class ConcreteStateDown : State {
7
8      int chargeTime = 0;
9      const int MAX_CHARGE = 10;
10

```

```

11     public override void DoAction(MyState state)
12     {
13         Debug.Log("Down !!!");
14         // 옆드리기
15         StartCoroutine(HandleDown(state));
16         // 기모으기
17         StartCoroutine(HandleSkill());
18     }
19
20     IEnumerator HandleDown(MyState state)
21     {
22         transform.Rotate(Vector3.right * 90.0f);
23         transform.position = new Vector3(0, 0.5f, 0);
24         yield return null;
25     }
26
27     IEnumerator HandleSkill()
28     {
29         chargeTime = 0;
30         while (true)
31         {
32             chargeTime++;
33             if (chargeTime > MAX_CHARGE)
34             {
35                 chargeTime = 0;
36                 // 스킬 발동;
37                 GetComponent<MyAction8>().setActionType(MyState.STATE_SKILL);
38                 yield return null;
39             }
40
41             yield return new WaitForSeconds(0.1f);
42         }
43     }
44
45 }

```

ConcreteStateAttack.cs

```

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  // ConcreteState 클래스 : 내려찍기
6  public class ConcreteStateAttack : State {
7
8      public override void DoAction(MyState state)
9      {
10         Debug.Log("Down Attack !!!");
11         StartCoroutine(HandleAttack(state));
12     }
13
14     IEnumerator HandleAttack(MyState state)
15     {

```

```

16     transform.position = new Vector3(0, 0.2f, 0);
17     yield return new WaitForSeconds(0.1f);
18     transform.position = new Vector3(0, 1.2f, 0);
19     yield return new WaitForSeconds(0.1f);
20     transform.position = new Vector3(0, 0.2f, 0);
21     yield return new WaitForSeconds(0.1f);
22
23     transform.position = new Vector3(0, 1.0f, 0);
24
25     GetComponent<MyAction8>().setActionType(MyState.STATE_STANDING);
26     //state = STATE.STATE_STANDING;
27 }
28 }

```

ConcreteStateSkill.cs

```

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  // ConcreteState 클래스 : 기 모은 후 공격
6  public class ConcreteStateSkill : State {
7
8      public override void DoAction(MyState state)
9      {
10         Debug.Log("Forward Attack !!!");
11         StartCoroutine("HandleSkill");
12     }
13
14     IEnumerator HandleSkill()
15     {
16         transform.eulerAngles = new Vector3(0, 90, 0);
17         transform.position = new Vector3(0, 1.0f, 0);
18
19         transform.Translate(Vector3.forward * 3.0f);
20         yield return new WaitForSeconds(0.1f);
21         transform.Translate(Vector3.back * 1.0f);
22         yield return new WaitForSeconds(0.1f);
23         transform.Translate(Vector3.forward * 1.0f);
24         yield return new WaitForSeconds(0.1f);
25
26         transform.position = new Vector3(0, 1.0f, 0);
27
28         GetComponent<MyAction8>().setActionType(MyState.STATE_STANDING);
29     }
30 }

```

MyAction8.cs

```
1  using UnityEngine;
2  using System.Collections;
3
4  public enum MyState
5  {
6      STATE_STANDING,
7      STATE_JUMPING,
8      STATE_DUCKING,
9      STATE_DIVING,
10     STATE_SKILL
11 }
12
13 public class MyAction8 : MonoBehaviour {
14
15     private MyState state;
16
17     // Concrete클래스들의 접근점(추상 클래스)
18     private State myState;
19
20     // 상태 클래스 교환...
21     public void setActionType (MyState state) {
22
23         // 현재 상태 저장
24         this.state = state;
25
26         // 다양한 상태 중에 어떤 것을 가져와야 할 지 모르므로
27         // 추상 클래스를 대표로 해서 가져온다.
28         Component c = gameObject.GetComponent<State>() as Component;
29
30         if (c != null) {
31             Destroy(c);
32         }
33
34         switch (state)
35         {
36             case MyState.STATE_STANDING:
37                 myState = gameObject.AddComponent<ConcreteStateStand>();
38                 myState.DoAction(state);
39                 break;
40             case MyState.STATE_JUMPING:
41                 myState = gameObject.AddComponent<ConcreteStateJump>();
42                 myState.DoAction(state);
43                 break;
44             case MyState.STATE_DUCKING:
45                 myState = gameObject.AddComponent<ConcreteStateDown>();
46                 myState.DoAction(state);
47                 break;
48             case MyState.STATE_DIVING:
49                 myState = gameObject.AddComponent<ConcreteStateAttack>();
50                 myState.DoAction(state);
51                 break;
```

```

52         case MyState.STATE_SKILL:
53             myState = gameObject.AddComponent<ConcreteStateSkill>();
54             myState.DoAction(state);
55             break;
56         default:
57             break;
58     }
59 }
60
61 void Start ()
62 {
63     setActionType(MyState.STATE_STANDING);
64 }
65
66 void Update()
67 {
68     switch (state)
69     {
70         case MyState.STATE_STANDING:
71             if (Input.GetKeyDown(KeyCode.Space))
72             {
73                 setActionType(MyState.STATE_JUMPING);
74             }
75             else if (Input.GetKeyDown(KeyCode.DownArrow))
76             {
77                 setActionType(MyState.STATE_DUCKING);
78             }
79             break;
80         case MyState.STATE_JUMPING:
81             if (Input.GetKeyDown(KeyCode.DownArrow))
82             {
83                 setActionType(MyState.STATE_DIVING);
84             }
85             break;
86         case MyState.STATE_DUCKING:
87             if (Input.GetKeyUp(KeyCode.DownArrow))
88             {
89                 setActionType(MyState.STATE_STANDING);
90             }
91             break;
92         default:
93             break;
94     }
95 }
96 }

```