State.cs

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

// State 추상클래스
public abstract class State : MonoBehaviour
{
    public abstract void DoAction(MyState state);
};

// 상태 인터페이스 클래스
//public interface State
//{
//  void DoAction();
//};
```

ConcreteStateStand.cs

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

// ConcreteState 클래스 : 서기
public class ConcreteStateStand : State {

    public override void DoAction(MyState state)
    {
        Debug.Log("Stand !!!");
        StartCoroutine(HandleStand(state));
    }

    IEnumerator HandleStand(MyState state)
    {
        transform.eulerAngles = new Vector3(0, 90, 0);
        transform.position = new Vector3(0, 1.0f, 0);
        yield return null;
    }
}
```

ConcreteStateJump.cs

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

// ConcreteState 클래스 : 점프
public class ConcreteStateJump : State
{
    float gravity = 0.0f;   // 중력의 값
    Vector3 velocity;       // 캐릭터의 현재 높이 저장값
    const int MAX_CHARGE = 20;

```

```
    public override void DoAction(MyState state)
    {
        Debug.Log("Jump !!!");
        velocity = transform.position;
        StartCoroutine(HandleJump(state));
    }

    IEnumerator HandleJump(MyState state)
    {
        gravity = 0.7f;

        while (true)
        {
            if (state == MyState.STATE_DIVING)
            {
                break;
            }

            velocity.y = velocity.y + gravity;

            transform.position = velocity;

            if (velocity.y < 1.0f)
            {
                break;
            }

            gravity = gravity - 0.1f;

            yield return new WaitForSeconds(0.05f);
        }

        gravity = 0.0f;
        velocity.y = 1.0f;
        transform.position = velocity;
        GetComponent<MyAction8>().setActionType(MyState.STATE_STANDING);
        // 위 코드 대신 점프 후 서 있는 상태 하나를 더 만들 수도 있다.
        //state = STATE.STATE_STANDING2;

        yield return null;
    }
}
```

ConcreteStateDown.cs

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

// ConcreteState 클래스 : 엎드리기
public class ConcreteStateDown : State {

    int chargeTime = 0;
    const int MAX_CHARGE = 10;

```

```
    public override void DoAction(MyState state)
    {
        Debug.Log("Down !!!");
        // 엎드리기
        StartCoroutine(HandleDown(state));
        // 기모으기
        StartCoroutine(HandleSkill());
    }

    IEnumerator HandleDown(MyState state)
    {
        transform.Rotate(Vector3.right * 90.0f);
        transform.position = new Vector3(0, 0.5f, 0);
        yield return null;
    }

    IEnumerator HandleSkill()
    {
        chargeTime = 0;
        while (true)
        {
            chargeTime++;
            if (chargeTime > MAX_CHARGE)
            {
                chargeTime = 0;
                // 스킬 발동;
                GetComponent<MyAction8>().setActionType(MyState.STATE_SKILL);
                yield return null;
            }

            yield return new WaitForSeconds(0.1f);
        }
    }

}
```

ConcreteStateAttack.cs

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

// ConcreteState 클래스 : 내려찍기
public class ConcreteStateAttack : State {

    public override void DoAction(MyState state)
    {
        Debug.Log("Down Attack !!!");
        StartCoroutine(HandleAttack(state));
    }

    IEnumerator HandleAttack(MyState state)
    {
```

```
        transform.position = new Vector3(0, 0.2f, 0);
        yield return new WaitForSeconds(0.1f);
        transform.position = new Vector3(0, 1.2f, 0);
        yield return new WaitForSeconds(0.1f);
        transform.position = new Vector3(0, 0.2f, 0);
        yield return new WaitForSeconds(0.1f);

        transform.position = new Vector3(0, 1.0f, 0);

        GetComponent<MyAction8>().setActionType(MyState.STATE_STANDING);
        //state = STATE.STATE_STANDING;
    }
}
```

ConcreteStateSkill.cs

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

// ConcreteState 클래스 : 기 모은 후 공격
public class ConcreteStateSkill : State {

    public override void DoAction(MyState state)
    {
        Debug.Log("Forward Attack !!!");
        StartCoroutine("HandleSkill");
    }

    IEnumerator HandleSkill()
    {
        transform.eulerAngles = new Vector3(0, 90, 0);
        transform.position = new Vector3(0, 1.0f, 0);

        transform.Translate(Vector3.forward * 3.0f);
        yield return new WaitForSeconds(0.1f);
        transform.Translate(Vector3.back * 1.0f);
        yield return new WaitForSeconds(0.1f);
        transform.Translate(Vector3.forward * 1.0f);
        yield return new WaitForSeconds(0.1f);

        transform.position = new Vector3(0, 1.0f, 0);

        GetComponent<MyAction8>().setActionType(MyState.STATE_STANDING);
    }
}
```

MyAction8.cs

```
using UnityEngine;
using System.Collections;

public enum MyState
{
    STATE_STANDING,
    STATE_JUMPING,
    STATE_DUCKING,
    STATE_DIVING,
    STATE_SKILL
}

public class MyAction8 : MonoBehaviour {

    private MyState state;

    // Concrete클래스들의 접근점(추상 클래스)
    private State myState;

    // 상태 클래스 교환...
    public void setActionType (MyState state) {

        // 현재 상태 저장
        this.state = state;

        // 다양한 상태 중에 어떤 것을 가져와야 할 지 모르므로
        // 추상 클래스를 대표로 해서 가져온다.
        Component c = gameObject.GetComponent<State>() as Component;

        if (c != null) {
            Destroy(c);
        }

        switch (state)
        {
            case MyState.STATE_STANDING:
                myState = gameObject.AddComponent<ConcreteStateStand>();
                myState.DoAction(state);
                break;
            case MyState.STATE_JUMPING:
                myState = gameObject.AddComponent<ConcreteStateJump>();
                myState.DoAction(state);
                break;
            case MyState.STATE_DUCKING:
                myState = gameObject.AddComponent<ConcreteStateDown>();
                myState.DoAction(state);
                break;
            case MyState.STATE_DIVING:
                myState = gameObject.AddComponent<ConcreteStateAttack>();
                myState.DoAction(state);
                break;
```

```
            case MyState.STATE_SKILL:
                myState = gameObject.AddComponent<ConcreteStateSkill>();
                myState.DoAction(state);
                break;
            default:
                break;
        }
    }

    void Start ()
    {
        setActionType(MyState.STATE_STANDING);
    }

    void Update()
    {
        switch (state)
        {
            case MyState.STATE_STANDING:
                if (Input.GetKeyDown(KeyCode.Space))
                {
                    setActionType(MyState.STATE_JUMPING);
                }
                else if (Input.GetKeyDown(KeyCode.DownArrow))
                {
                    setActionType(MyState.STATE_DUCKING);
                }
                break;
            case MyState.STATE_JUMPING:
                if (Input.GetKeyDown(KeyCode.DownArrow))
                {
                    setActionType(MyState.STATE_DIVING);
                }
                break;
            case MyState.STATE_DUCKING:
                if (Input.GetKeyUp(KeyCode.DownArrow))
                {
                    setActionType(MyState.STATE_STANDING);
                }
                break;
            default:
                break;
        }
    }
}
```