Unity3D Script

Coroutine

코드

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


Unity 스크립트|참조 0개
public class corutine : MonoBehaviour
{
    public bool isDelay;
    public float delayTime = 5.0f;
    public float accumTime;

    Unity 메시지|참조 0개
    private void Update()
    {
        if (Input.GetKeyDown(KeyCode.Space))
        {
            if(isDelay == false)
            {
                isDelay = true;
                Debug.Log("HP가 50 회복 되었습니다.");
            }
            else
            {
                Debug.Log("아직 포션을 사용할 수 없습니다. ");
            }
        }
        if(isDelay)
        {

            accumTime += Time.deltaTime;
            if(accumTime >= delayTime)
            {

                accumTime = 0.0f;
                isDelay = false;


            }
        }
    }


}
```

코드

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

Unity 스크립트|참조 0개
public class corutine_with : MonoBehaviour
{
    public bool isDelay;
    public float delayTime = 5.0f;
    public float accumTime;
    private float interface_value = 0.0f;
    private int counter = 0;

    Unity 메시지|참조 0개
    private void Update()
    {
        ++counter;

        if (Input.GetKeyDown(KeyCode.A))
        {
            interface_value = interface_value + 1.0f;
            Debug.Log(interface_value);
        }
        else if (Input.GetKeyDown(KeyCode.Space))
        {
            if (isDelay == false)
            {
                isDelay = true;
                Debug.Log("HP가 50 회복 되었습니다.");
                StartCoroutine(Drink(interface_value));
            }
            else
            {
                Debug.Log("아직 포션을 사용할 수 없습니다. ");

            }
        }
        else if(Input.GetKeyDown(KeyCode.D))
        {
            StopCoroutine("Drink");
        }

        //if(counter % 100 == 0) Debug.Log("카운터" + counter);


    }
```

```
    참조 1개
    IEnumerator Drink(float test_value)
    {
        Debug.Log("코루틴시작");
        //yield return new WaitForSeconds(5.0f);
        //yield return new WaitForEndOfFrame();
        //yield return new WaitForFixedUpdate();

        //다음 프레인 바로 시작
        //yield return null;

        //바로 종료 (밑에 코드 실행 안됨)
        //yield break;

        Debug.Log("코루틴이 받은 값" + test_value);
        yield return new WaitUntil( ()=> test_value > 3) ;

        Debug.Log("코루틴 안 변수 출력"+test_value);
        isDelay = false;
    }
}
```

내용

- 프로그래밍 동시성 vs 병행성
- 프로그래밍 동기 vs 비동기 처리
  - https://velog.io/@gogumi4502/Kotlin-%EB%8F%99%EA%B8%B0-%EB%B9%84%EB%8F%99%EA%B8%B0-%EC%98%88%EC%99%B8%EC%B2%98%EB%A6%AC
- C++ 반복자/코루틴
  - https://blog.naver.com/jhsh8788/221372280171
- 파이썬 반복자/코루틴
  - https://dojang.io/mod/page/view.php?id=2418
  - https://miki3079.tistory.com/112
- C# 반복자/코루틴
- 유니티 C# 반복자/코루틴
- 유니티 동기/비동기 프로그래밍
  - https://velog.io/@ljun970/%EC%BD%94%EB%A3%A8%ED%8B%B4%EA%B3%BC-%EB%B9%84%EB%8F%99%EA%B8%B0-%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%98%EB%B0%8D
  - https://ljhyunstory.tistory.com/284