UNITY

Intermediate

Invoke, Coroutine

함수에 딜레이 넣기

개발을 하다가, 함수에 딜레이를 넣고 싶을 때가 많을 겁니다. 예를들어 30초 뒤에 게임이 시작되게 하고 싶을수도 있고, 함수 내부 자체에서 다음 코드를 실행하기 전에 몇초간 기다리게 하고 싶을 수도 있습니다.

물론 실제로 몇초가 걸렸는지 계산해서 이 기능을 구현할 수도 있지만, 유니티에서는 이미 이 기능들을 구현하고 있습니다! 그럼 한번 알아보도록 하죠.

Invoke 사용하기

Invoke 메서드는 특정 함수를 일정 시간 후에 호출되게 만들고 싶을 때 사용합니다.

Invoke(string name, float time)

과 같이 사용하며, 메서드의 **이름** 과 원하는 지연시간을 입력받습니다. 이름으로 호출하기 때문에 매개변수가 있을 수가 없으며, 공식 예제는 다음과 같습니다.

2초 뒤에 LaunchProjectile() 를 호출합니다.

```
Rigidbody projectile;

void Start()
{
    Invoke("LaunchProjectile", 2.0f);
}

void LaunchProjectile()
{
    Rigidbody instance = Instantiate(projectile);
    instance.velocity = Random.insideUnitSphere * 5.0f;
}
```

InvokeRepeating

만약에 한번만 호출하는 것이 아니라, 계속해서 반복적으로 호출하고 싶다면 InvokeRepeating을 사용할 수 있습니다.

InvokeRepeating(string n, float t, float repeatRate)

Invoke에 매개변수가 하나 추가되었는데, t만큼 기다린 이후부터 몇초 간격으로 반복해서 호출할지를 정해줍니다. 사용법은 거의 동일하기때문에 생략하겠습니다.

CancelInvoke

현재 MonoBehavior에서 실행한 모든 Invoke를 취소합니다. InvokeRepeating으로 설정된 것을 해제할 때 주로 사용하지만, 걸어둔 Invoke도 가끔 취소할 경우가 생깁니다.

```
CancelInvoke()
```

코루틴Coroutine

그런데 사실 성능문제도 있고, 유지보수적으로 좋지 않기 때문에(함수 이름이 변경되면...?) Invoke 대신 훨씬 강력한 코루틴Coroutine 을 주로 사용하게 됩니다.

```
void Start()
{
    StartCoroutine(Move());
}

참조 1개
IEnumerator Move()
{
    while (true)
    {
        transform.position += transform.right;
        yield return new WaitForSeconds(1f);
    }
}
```

위의 코드는 1초마다 오른쪽으로 한칸씩 움직이게 해줍니다. 여기서 주목해야 할 점은 IEnumerator 타입과 yield return입니다.

Coroutine 관리함수

StartCoroutine(IEnumerator routine)

방금 전의 예제에서 봤듯이 코루틴(IEnumerator 타입) 함수 호출문을 넣어서 호출하며, 매개변수도 넣어줄 수 있습니다.

StopCoroutine(IEnumerator routine)

매개변수로 IEnumerator 또는 Coroutine 타입을 받는데, 아래의 예제 코드를 참고해보세요.

```
void Start()
{
    IEnumerator coroutine = Move();
    StartCoroutine(coroutine);
    StopCoroutine(coroutine);

    Coroutine routine = StartCoroutine(Move());
    StopCoroutine(routine);
}
```

YieldInstruction

yield return이란, 함수의 중간 값을 리턴한다는 말입니다. 실행을 지연시킨다고 할 수 있죠. yield return null 을 하면 한 프레임만큼 기다리지만, 우리는 주로 특정 시간만큼 기다리기를 원하기 때문에 이를 위한 클래스들이 존재합니다.

여기서 주로 사용되는 것은 예제에서도 나왔던 클래스로, 원하는 초만큼 기다려줍니다.

yield return new WaitForSeconds(float time)

또는, 코루틴 자체를 리턴해서 해당 코루틴이 끝날때 까지 기다리게 해줄수도 있습니다.

yield return StartCoroutine(IEnumerator routine);

WaitForSeconds - Unity Scripting API

코루틴은 많이 사용되고, 매우 유용하기 때문에 익숙하게 다룰 수 있도록 많이 실습해봅시다. 또한, 위의 개념들만 알아도 개발에 아무 문제가 없지만 추가 기능을 알고싶으시다면 유니티 공식 API를 참고해보세요.