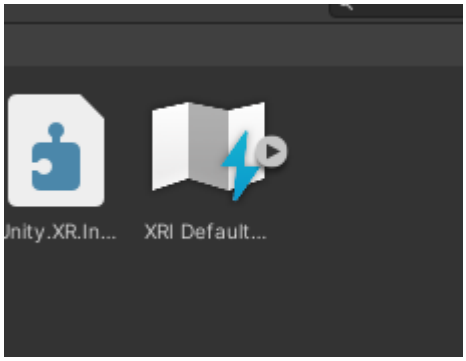
Action Based 컨트롤러 로직 작성 방법

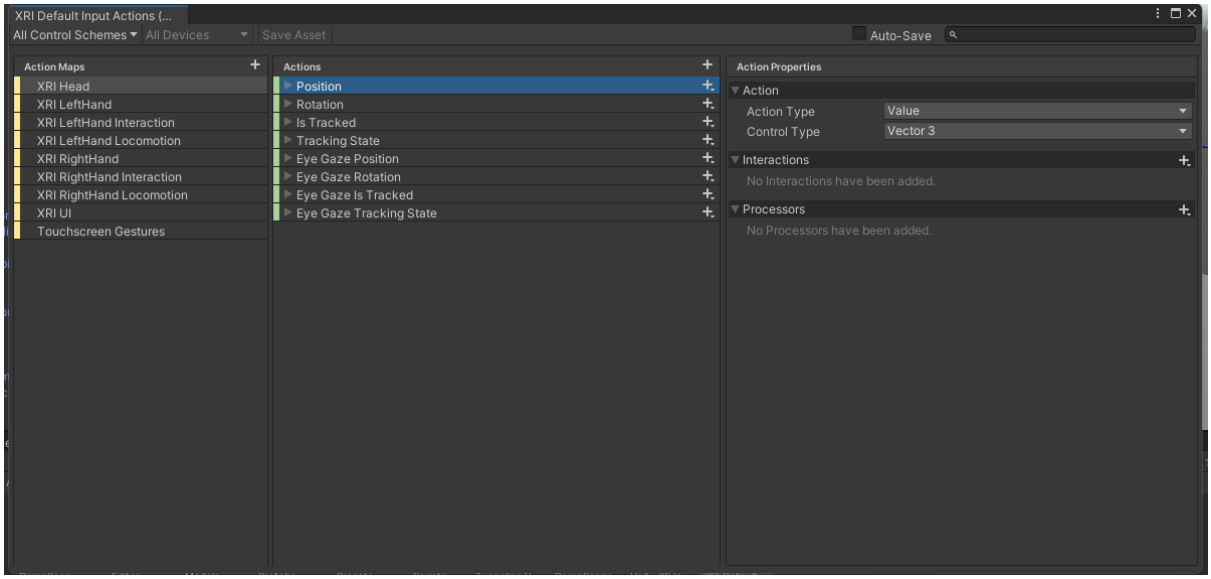
1. 사용하고 싶은 액션 등록 (기본적으로 있는거 쓰려면 2로)

Assets/Samples/XR Interaction Toolkit/2.5.4/Starter Assets/XRI Default Input Actions.inputactions

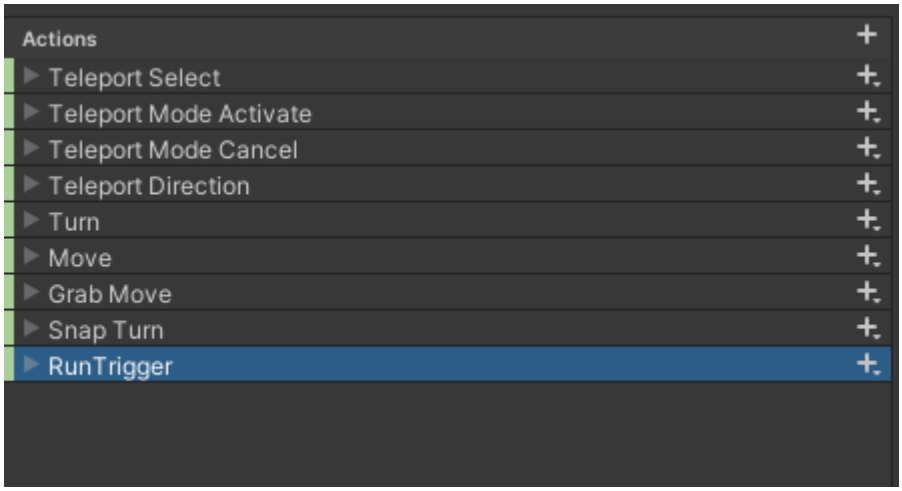
- 더블 클릭



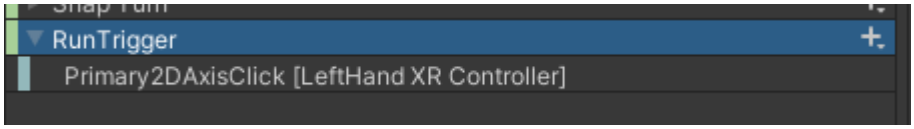
- 원하는 Action Maps 선택(만들어도 되지만 그냥 있는거 써도 될 것 같다.)



+ 눌러서 새 액션 등록

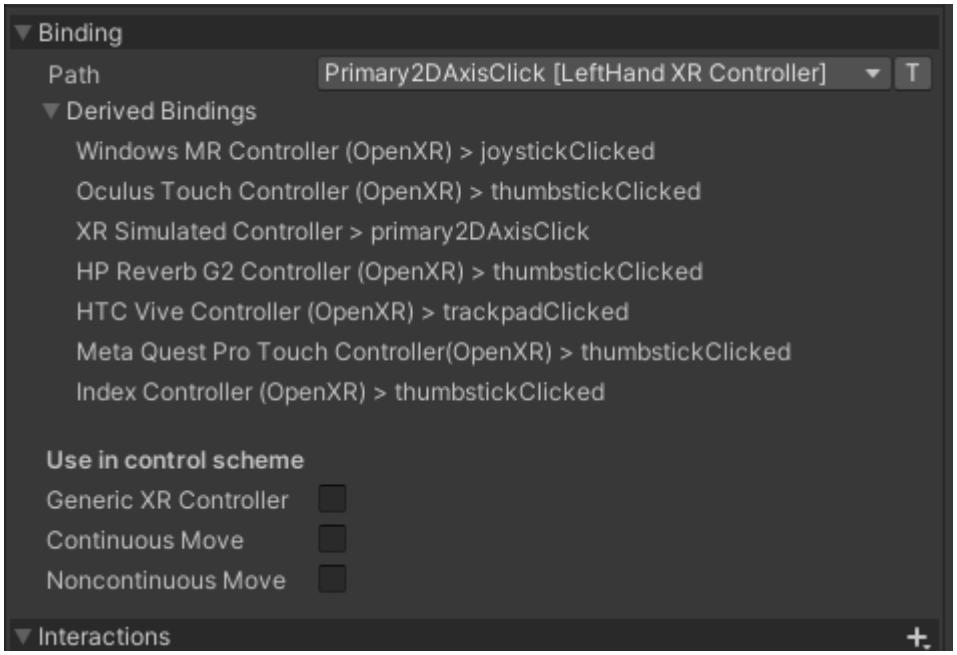


만든 액션에 + 눌러서 Add Binding



Path에 컨트롤러의 원하는 부분 등록(컨트롤러 말고 키보드도 가능하다.)

(XR Interaction ToolKit 기능이 아니라 Unity 기본 Input Action 기능이기 때문)



필요하다면 Interactions에 세부 상호작용 방식을 정한다

Hold -> 정한 시간만큼 누르고 있는다.

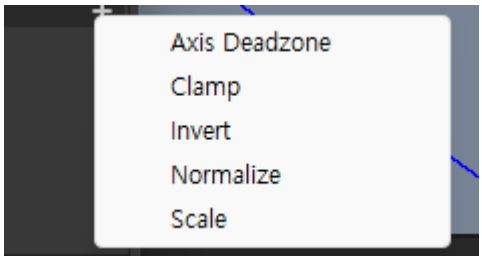
Multi Tap -> 일정 시간 내에 n번 눌렀다 뗀다.

Press -> 누른다, 누르기만 할건지 누르고 뗄건지 정한다.

Slow Tap -> 일전 시간 만큼 이상 눌렀다가 뗀다

Tap -> 일정 시간 내로 눌렀다가 뗀다.

필요하다면 Processors에서 값을 리턴할 수 있도록 + 보정을 정해준다.



사용자가 컨트롤러 조작을 했을 때 값을 읽어들일 수 있다.

(잘 안쓸 것 같다.)

저장한다.

2. 스크립트 제작

```
Unity 스크립트(자산 참조 1개)|참조 0개
public class ContinuousRunTrigger : MonoBehaviour {

    [Header("Action Variable")]
    [SerializeField, Tooltip("Register the action you want.")] private InputActionReference actionReference;
    [SerializeField, Tooltip("This function is used only on continuous movement.")] private  ContinuousMoveProviderBase cmBase;
    [SerializeField, Tooltip("This value is multiplied by the basic moving speed.")] private float mulSpeed;

    // event register
    Unity 메시지|참조 0개
    private void OnEnable() {
        actionReference.action.performed += SetRun;
        actionReference.action.canceled += SetOrigin;
    }

    // event unregister
    Unity 메시지|참조 0개
    private void OnDisable() {
        actionReference.action.performed -= SetRun;
        actionReference.action.canceled -= SetOrigin;
    }
```

```
참조 2개
private void SetRun(InputAction.CallbackContext obj) {

    if (mulSpeed == 0) {
        Define.LogError("multiplied value is zero");
        return;
    }

    if (cmBase == null) {
        Define.LogError("Move Provider is null");
        return;
    }

    cmBase.moveSpeed *= mulSpeed;
}

참조 2개
private void SetOrigin(InputAction.CallbackContext obj) {

    if (mulSpeed == 0) {
        Define.LogError("multiplied value is zero");
        cmBase.moveSpeed = 0;
        return;
    }

    if (cmBase == null) {
        Define.LogError("Move Provider is null");
        return;
    }

    cmBase.moveSpeed /= mulSpeed;
}
```

이런식으로 액션을 직렬화 해서 초기화한 다음 함수를 등록해준다.

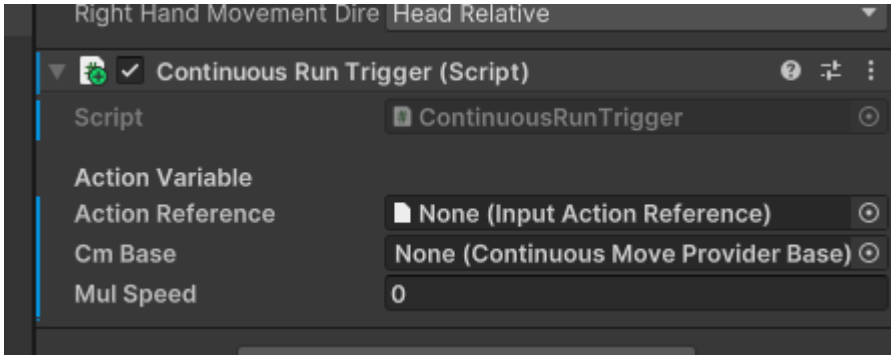
started -> 액션을 시도했을 때

performed -> 실제 액션이 수행될 때

canceled -> 액션이 취소될 때

예를 들어서 0.5초동안 지속적으로 누르고 있어야 하는 액션이라면

started -> 0.5s -> performed -> 손에서 버튼 뗌 -> canceled 와 같은 방식이다.



만든 스크립트를 붙이고, 액션을 등록해주면 끝

참고 자료

[Input System | Input System | 1.8.2 (unity3d.com)]