가상현실 및 실습

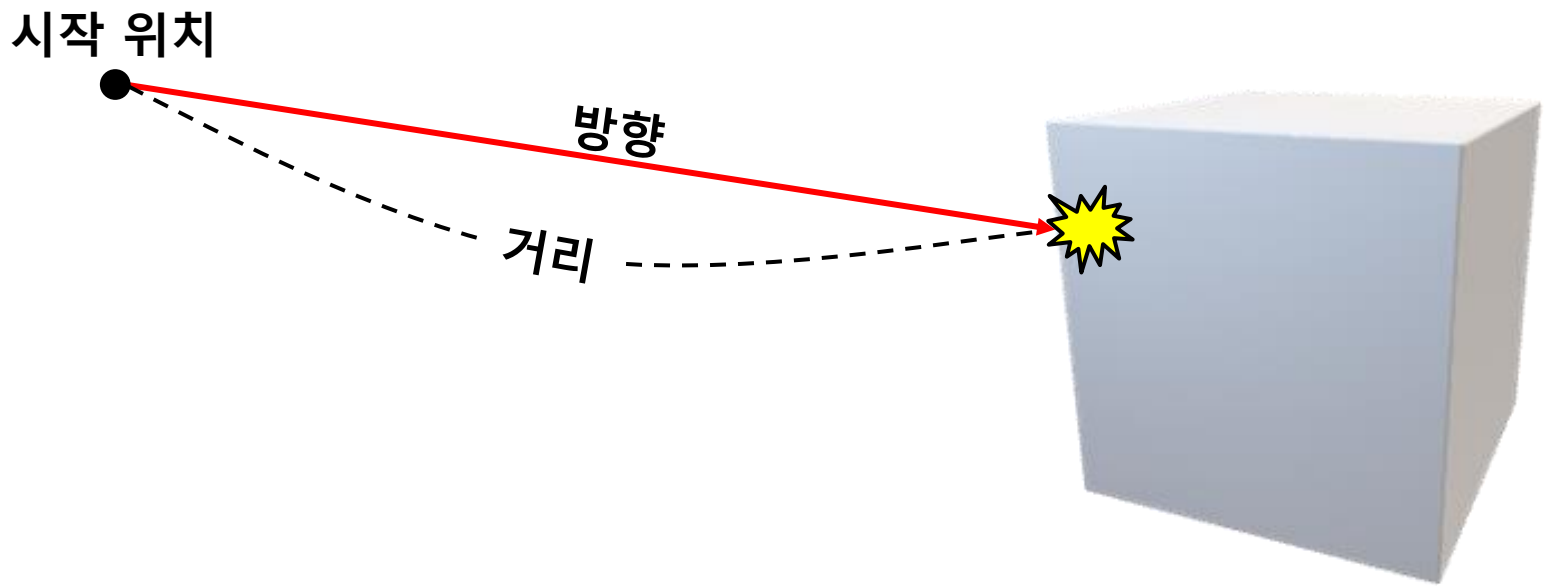
레이캐스트

레이 케스트 활용

1. 레이 캐스트 활용

1) 레이 및 레이 캐스트

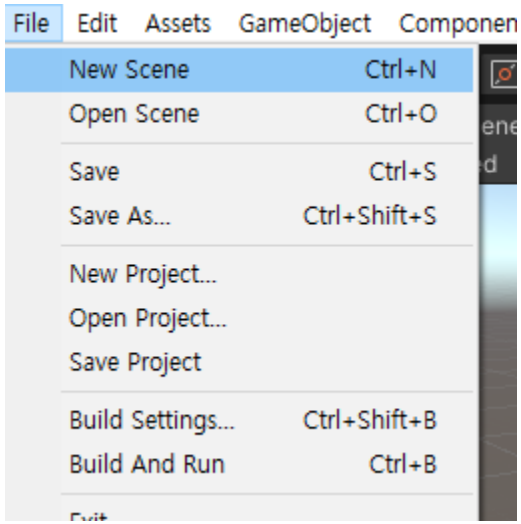
- 레이 캐스트(Raycast)는 충돌할 수 있는 광선(Ray)을 쏘는 것의 의미함
- 레이는 시작 위치, 방향, 거리 정보를 바탕으로 생성됨
- 레이 캐스트를 수행하였을 때, 충돌체와 부딪히면 충돌 정보를 받아올 수 있음

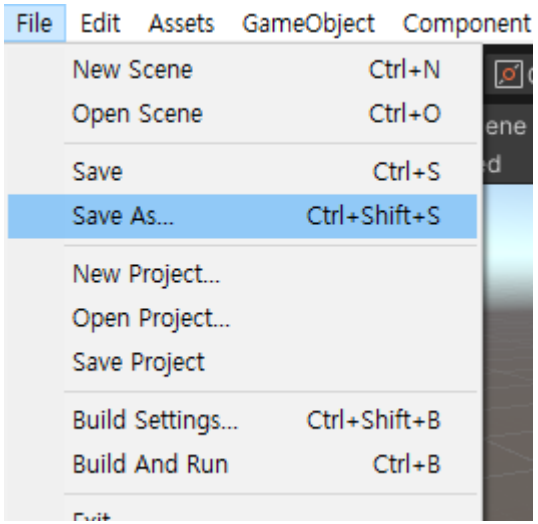


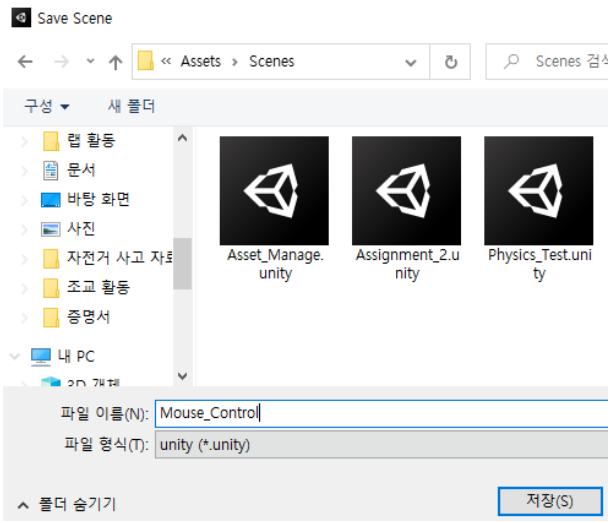
1. 레이 캐스트 활용

2) 씬 설정

- **메뉴 > File > New Scene을 선택**하여 새로운 씬을 생성함
- **메뉴 > File > Save As...를 선택**하여, 생성한 씬을 폴더 "Scenes"에 저장함
- **저장 시, 파일 이름은 “RayCastTest.unity"**





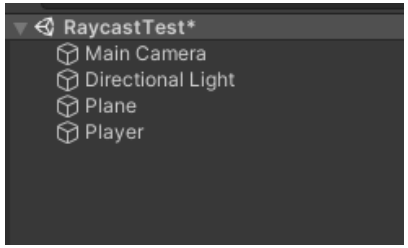


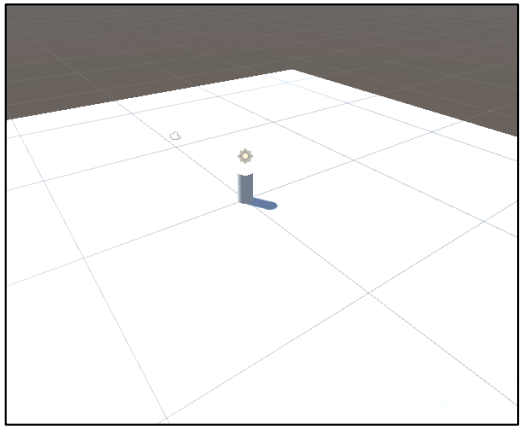
1. 레이 캐스트 활용

2) 씬 설정

- Hierarchy > 마우스 우 클릭 > 3D Object > Plane을 생성
- Hierarchy > 마우스 우 클릭 > 3D Object > Cylinder를 생성
- 생성된 **오브젝트 "Plane","Cylinder" Transform 값을 다음과 같이 설정**

	Plane	Cylinder
Name	**Plane**	Player
Position	**(0, 0, 0)**	**(0, 1, 0)**
Rotation	**(0, 0, 0)**	**(0, 0, 0)**
Scale	**(5, 5, 5)**	**(1, 1, 1)**



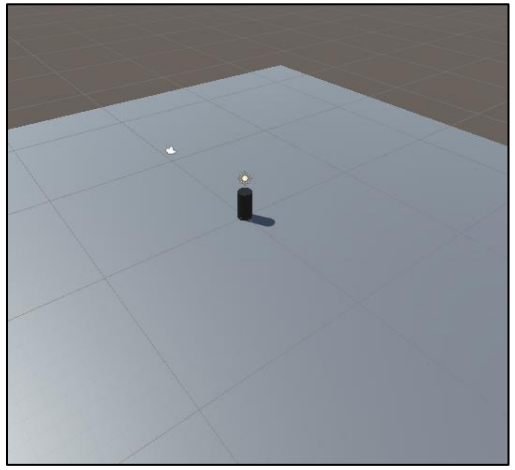


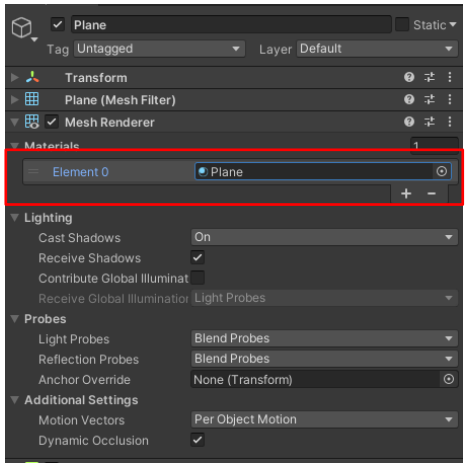
1. 레이 캐스트 활용

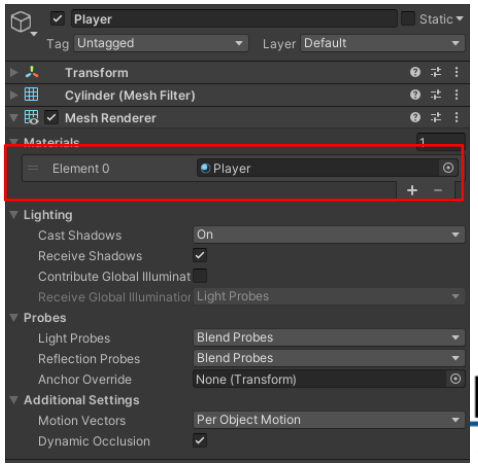
2) 씬 설정

- 프로젝트 창에 Material 폴더를 생성 후 2개의 Material을 생성
- Material의 이름을 각각 Plane_Mat, Player_Mat로 변경
- Plane_Mat는 Plane 오브젝트 컴포넌트로 추가, Player_Mat는 Player 오브젝트 컴포넌트로 추가
- 생성된 Material **"Plane_Mat","Player_Mat" RGBA값을 다음과 같이 설정**

	Plane_Mat	Playe_Mat
(R,G,B,A)	(103, 103, 103, 255)	(0, 0, 0, 255)



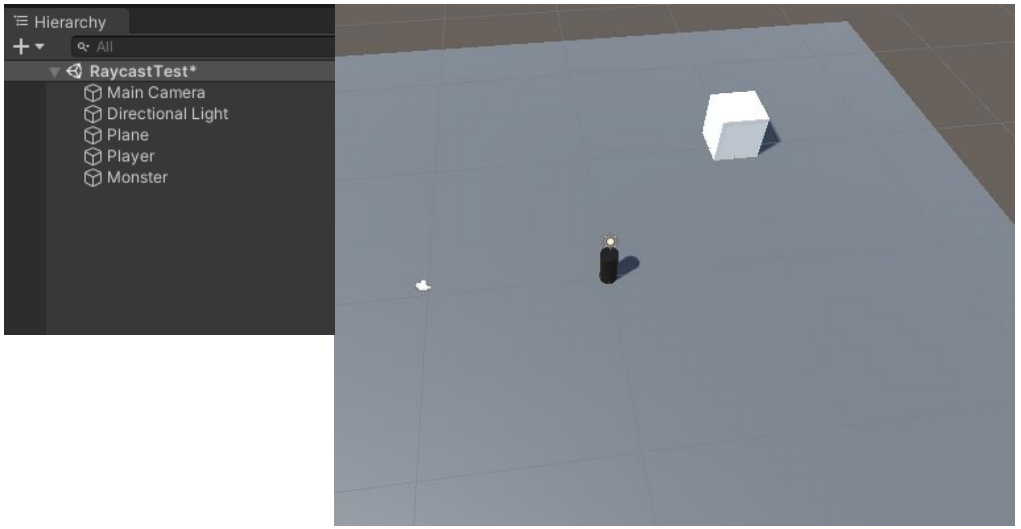




2) 씬 설정

- Hierarchy > 마우스 우 클릭 > 3D Object > Cylinder를 생성
- 생성된 **오브젝트 "Cube" 오브젝트를 "Monster"로 이름 변경**
- **"Monster" 오브젝트의 Transform 값을 다음과 같이 설정**

	Cube
Name	Monster
Position	(-10.25, 1.5, 10)
Rotation	(0, 0, 0)
Scale	(3, 3, 3)



1. 레이 캐스트 활용

3) 레이 캐스트 처리를 위한 스크립트

- 프로젝트 창에서 Scripts 폴더 생성 후 "PlayerRay" 스크립트 생성
- 아래와 같이 스크립트 작성
- Debug.DrawRay : 레이캐스트 광선을 Scene 창에 개발자가 확인할 수 있도록 그리는 기능

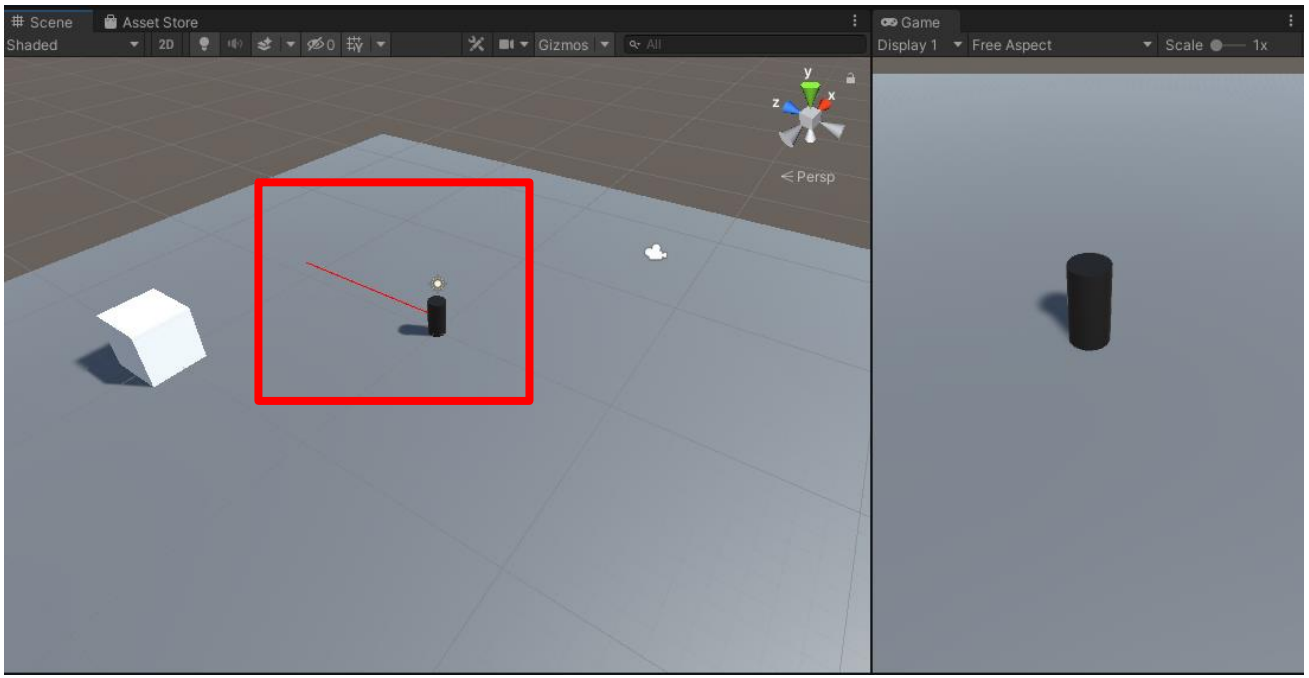
public static void DrawRay(Vector3 start, Vector3 dir, Color color = Color.white, float duration = 0.0f, bool depthTest = true)

start	레이를 시작하는 월드 공간에서의 지점.
dir	레이의 방향과 길이
color	라인의 색상
duration	라인을 표시할 시간(초).
depthTest	라인이 카메라에서 가까운 오브젝트에 의해 숨겨진 경우에 라인 숨기기 여부.

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

Unity 스크립트 | 참조 0개
public class PlayerRay : MonoBehaviour
{

    // Update is called once per frame
    Unity 메시지 | 참조 0개
    void Update()
    {
        Debug.DrawRay(this.transform.position, Vector3.forward * 10, Color.red);
    }
}
```

3) 레이 캐스트 처리를 위한 스크립트

- PlayerRay 스크립트를 Player 컴포넌트로 추가
- 게임 실행 시, Player 오브젝트를 기준으로 Z 방향으로 빨간 선이 보이는 것을 확인할 수 있음
- Debug.DrawRay는 Scene 창에서 확인 가능함. Game 창에서는 렌더링 되지 않음



1. 레이 캐스트 활용

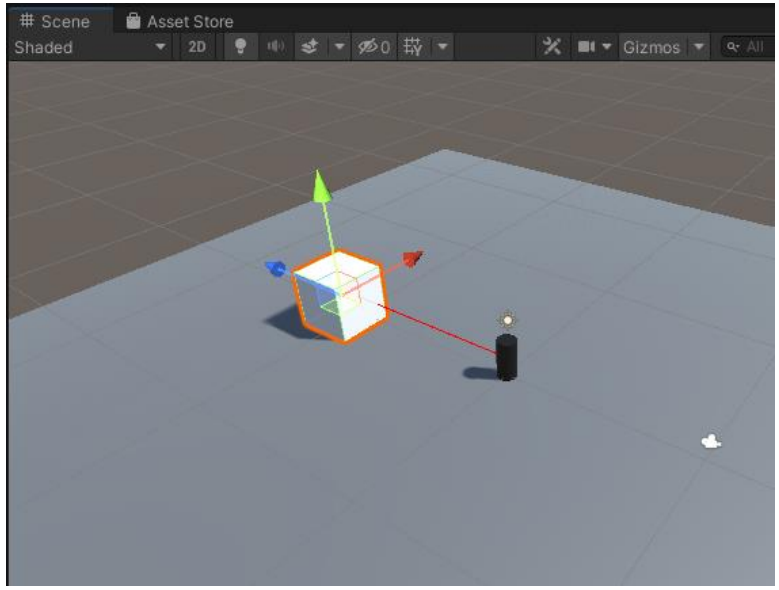
3) 레이 캐스트 처리를 위한 스크립트

- "PlayerRay" 스크립트에서 아래의 코드를 추가 작성
- RaycastHit : 레이캐스트 상에서 충돌한 오브젝트의 정보가 저장됨 (2D의 경우 RaycastHit2D)
- Physice.Raycast(시작점, 방향, 충돌 오브젝트 정보, 거리) : Ray를 발상하여 충돌한 오브젝트의 정보를 저장함

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

Unity 스크립트 | 참조 0개
public class PlayerRay : MonoBehaviour
{

    // Update is called once per frame
    Unity 메시지 | 참조 0개
    void Update()
    {
        Debug.DrawRay(this.transform.position, Vector3.forward * 10, Color.red);

        RaycastHit rayInfo;

        if(Physics.Raycast(this.transform.position, this.transform.forward, out rayInfo, 50.0f))
        {
            Debug.Log(rayInfo.transform.name);
        }
    }
}
```

3) 레이 캐스트 처리를 위한 스크립트

- 게임 실행 시, Ray 상에 오브젝트가 있는 경우 콘솔 창에 오브젝트의 이름을 확인할 수 있음



3) 레이 캐스트 처리를 위한 스크립트

- Monster 오브젝트의 Position을 (0, 1.5, 10)으로 변경
- Hierarchy > 마우스 우 클릭 > 3D Object > Cube를 생성
- Cube의 이름을 "Wall"로 변경
- 생성된 **오브젝트 " Wall" Transform 값을 다음과 같이 설정**

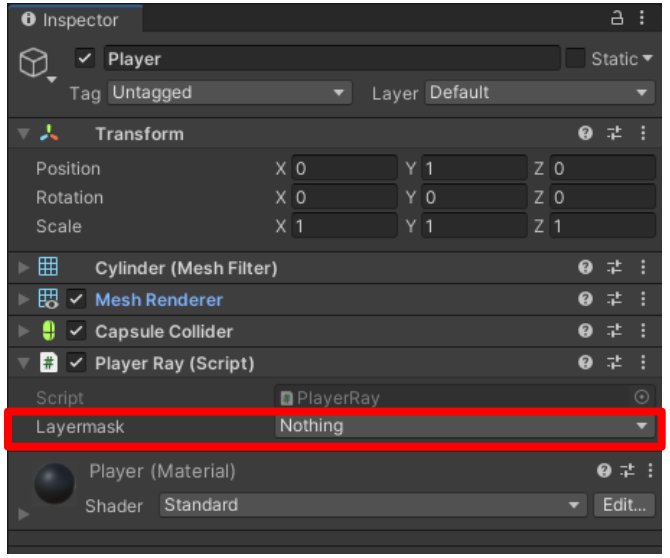
	Cube
Name	**Wall**
Position	**(0, 2.48, 4.5)**
Rotation	**(0, 0, 0)**
Scale	**(5, 5, 1)**

1. 레이 캐스트 활용

3) 레이 캐스트 처리를 위한 스크립트

- "PlayerRay" 스크립트에서 아래의 코드를 추가 작성
- LayerMask : 특정 레이어로 지정한 오브젝트를 선별적으로 확인할 때 사용됨

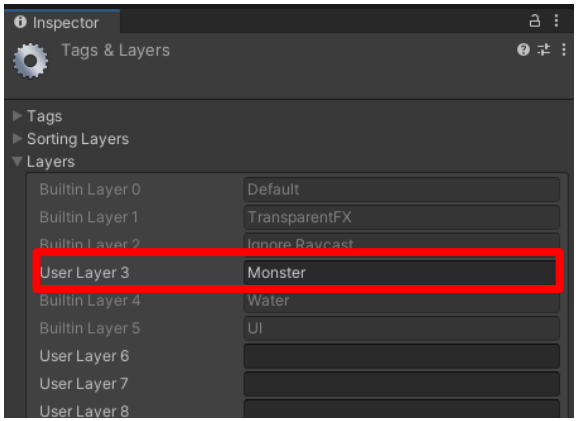
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

Unity 스크립트(자산 참조 1개)|참조 0개
public class PlayerRay : MonoBehaviour
{
    [SerializeField] LayerMask layermask;

    // Update is called once per frame
    Unity 메시지|참조 0개
    void Update()
    {
        Debug.DrawRay(this.transform.position, Vector3.forward * 10, Color.red);

        RaycastHit rayInfo;

        if(Physics.Raycast(this.transform.position, this.transform.forward, out rayInfo, 50.0f, layermask))
        {
            Debug.Log(rayInfo.transform.name);
        }
    }
```

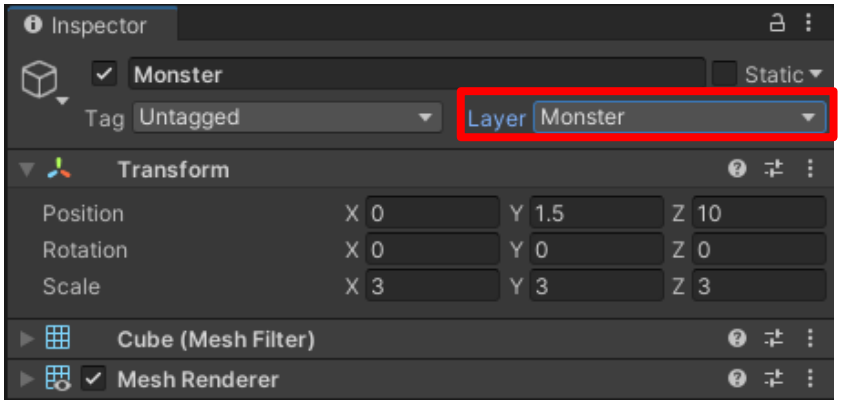


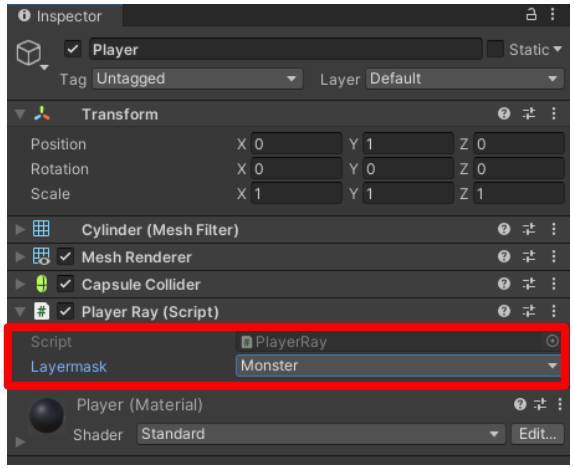
1. 레이 캐스트 활용

3) 레이 캐스트 처리를 위한 스크립트

- Inspector창에서 Layer에 AddLayer를 통해 "Monster" Layer로 추가
- Monster 오브젝트의 Layer를 Monster로 설정
- Player에 PlayerRay 컴포넌트의 Layermask를 Monster로 설정







1. 레이 캐스트 활용

3) 레이 캐스트 처리를 위한 스크립트

- 게임 실행 시, Wall 오브젝트는 무시
- 뒤에 있는 Monster 오브젝트가 콘솔 창에 출력되는 것을 확인