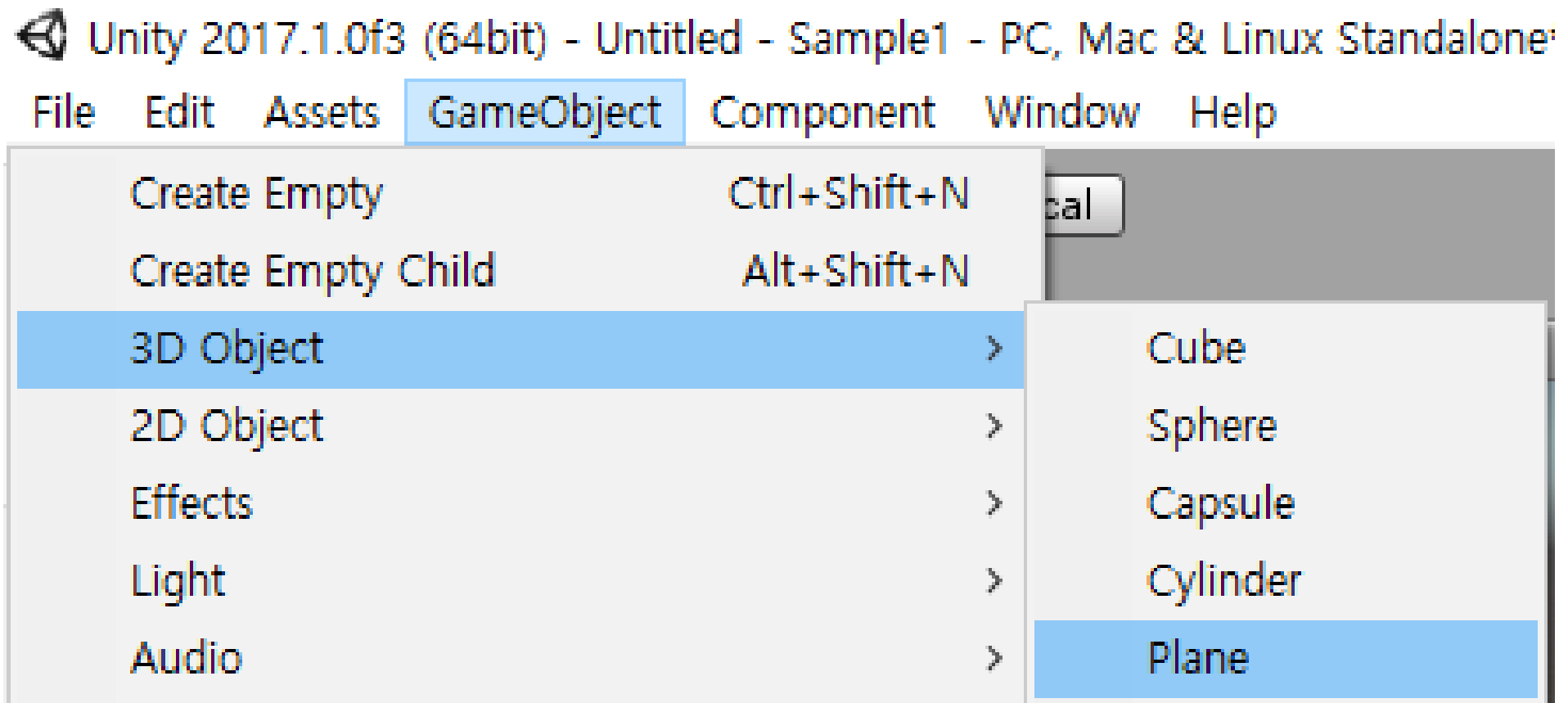


Unity – Component Basic

NHN NEXT
서형석

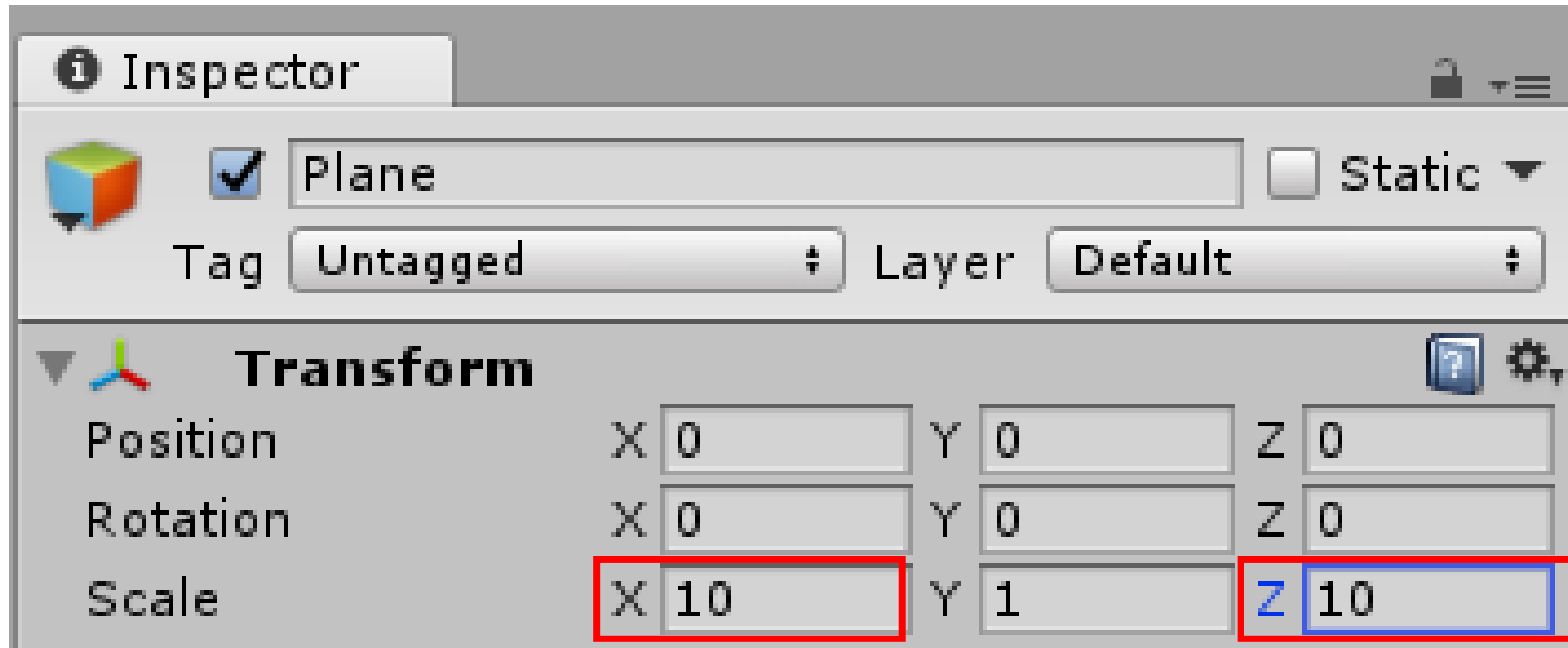
Unity

- 바닥(Plane) 구성



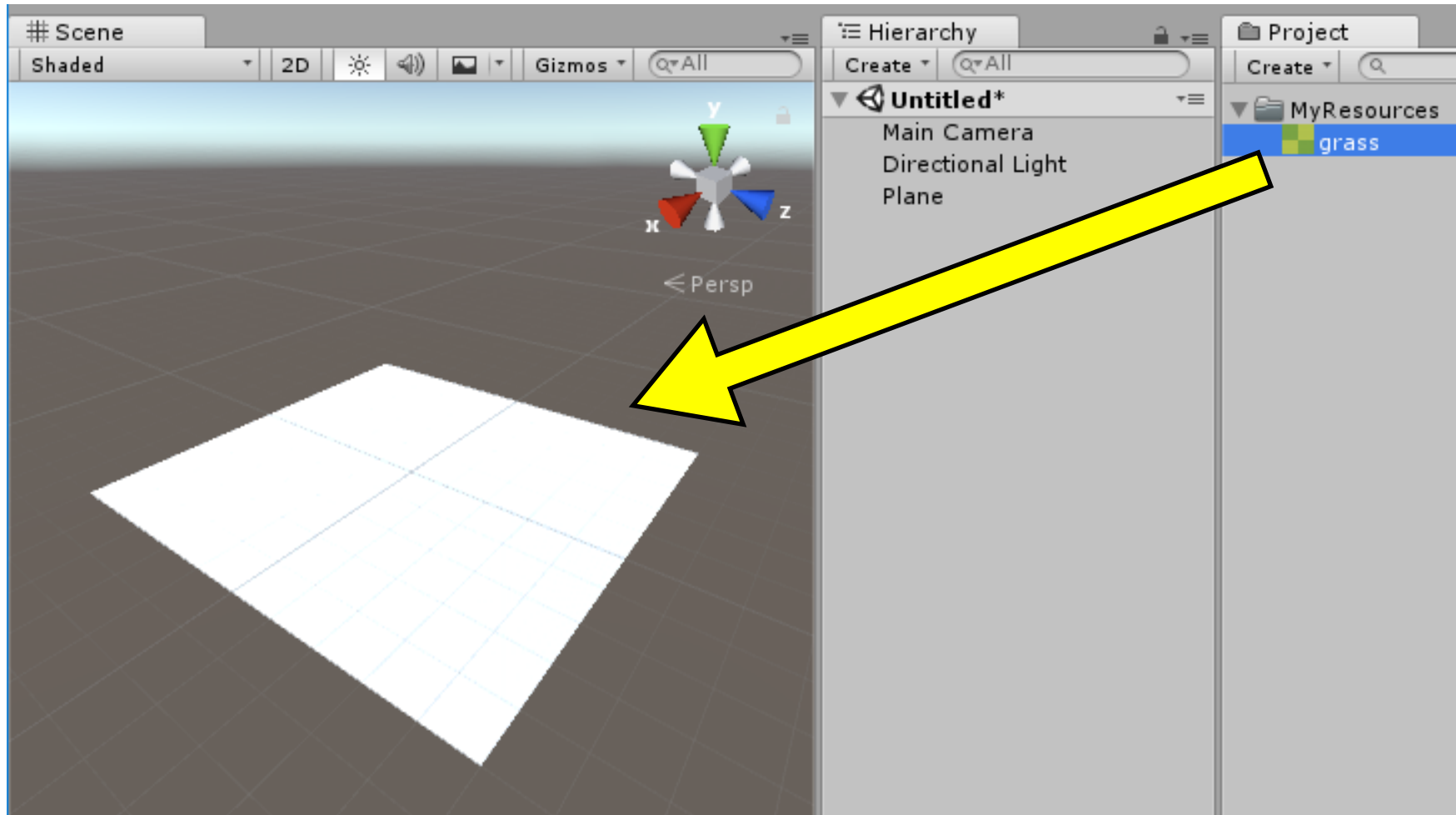
Unity

- 바닥 Transform Component 조절



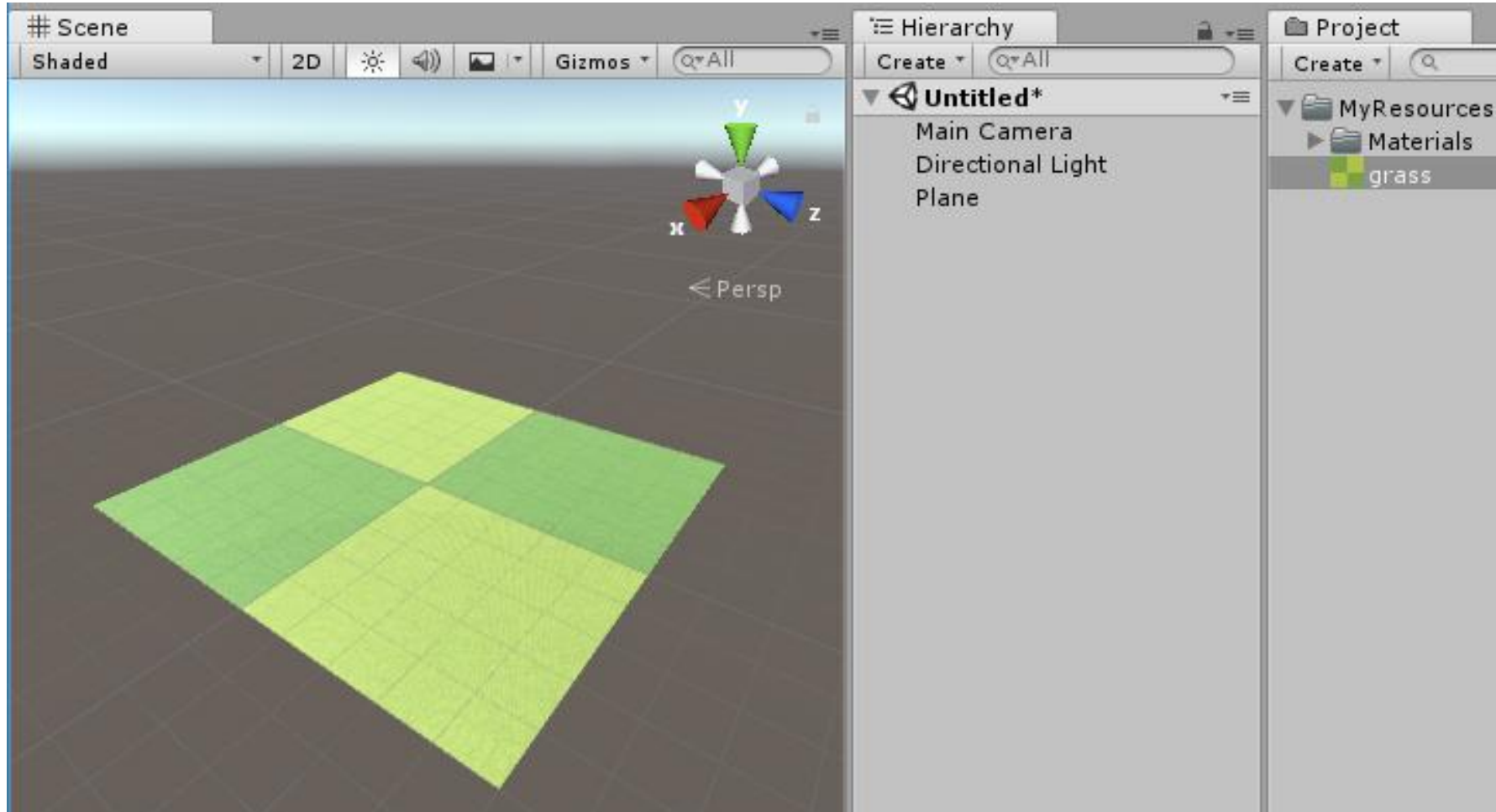
Unity

- grass Texture 바닥에 적용.



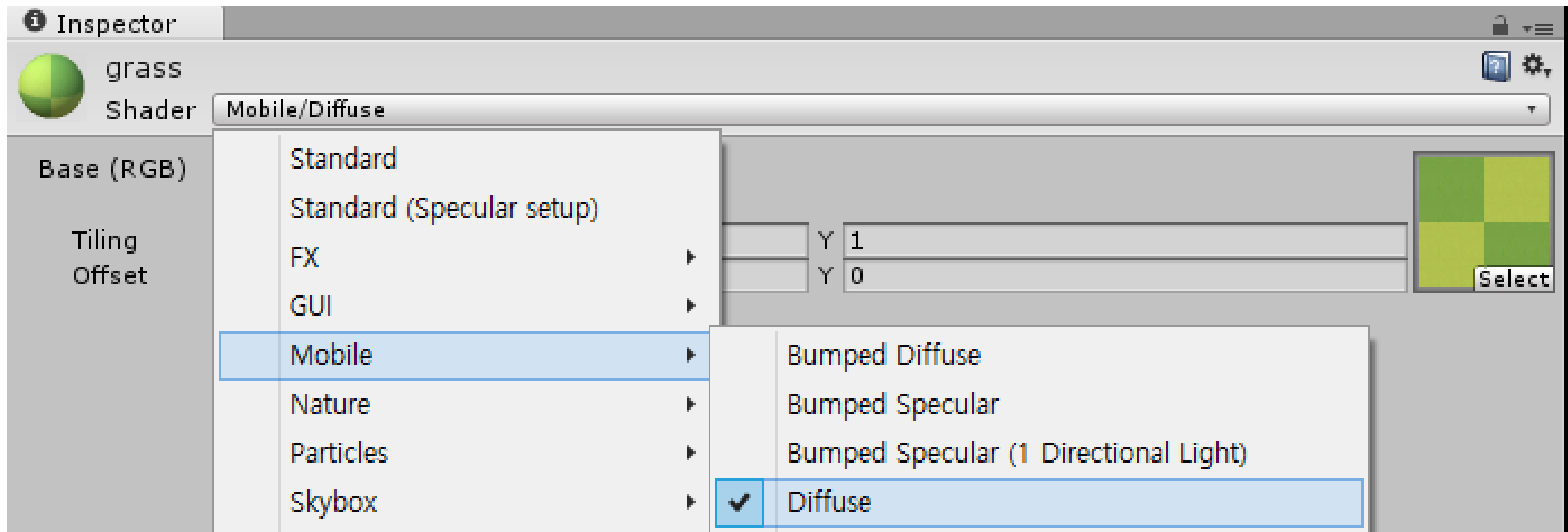
Unity

- grass Texture 바닥에 적용.



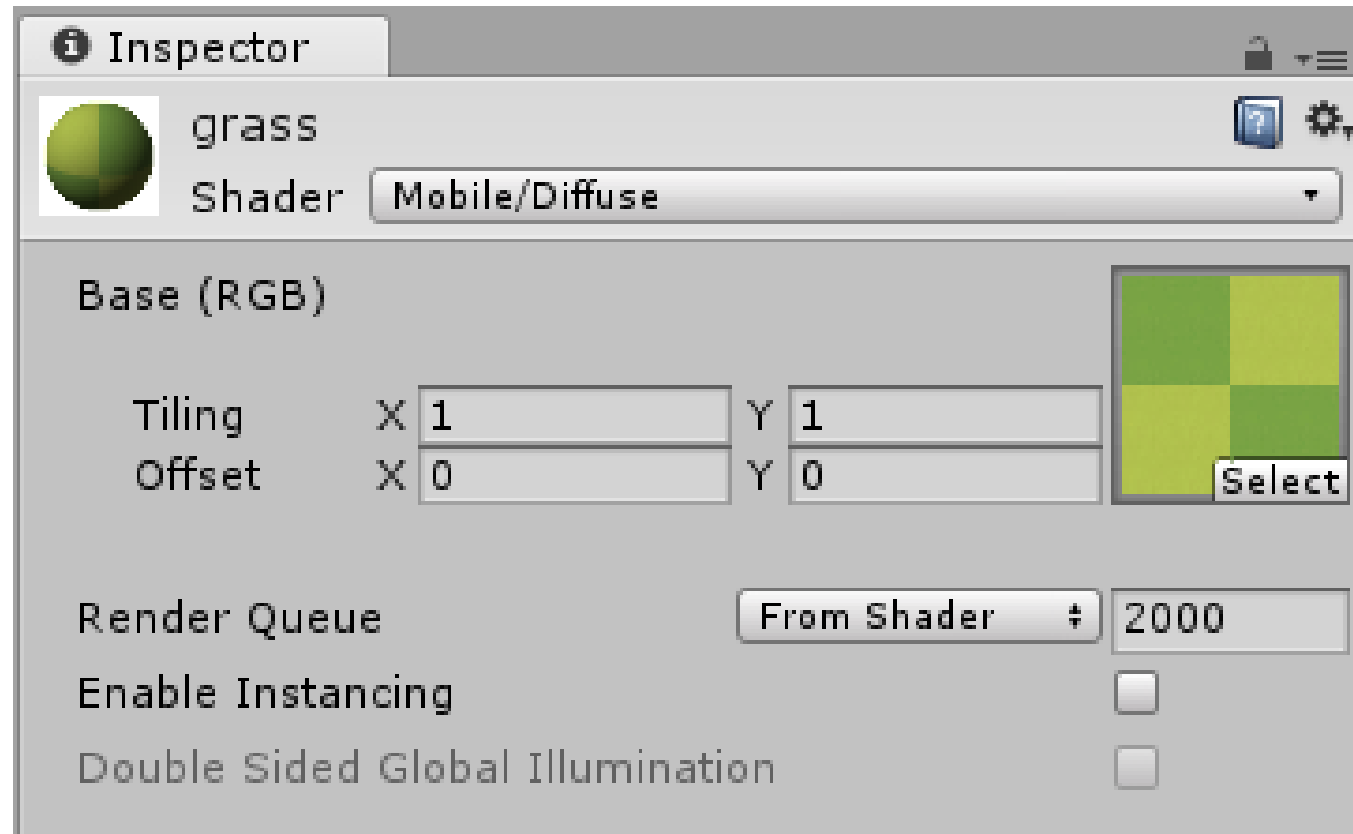
Unity

- Shader 교체



Unity

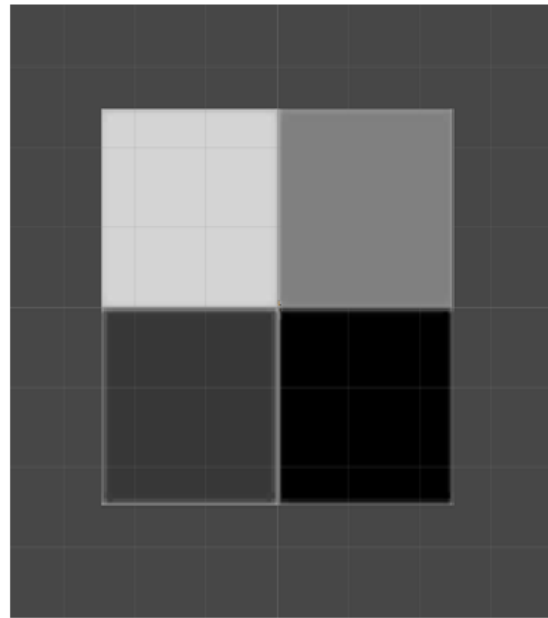
- Shader 교체 및 Tileling 수치 조절



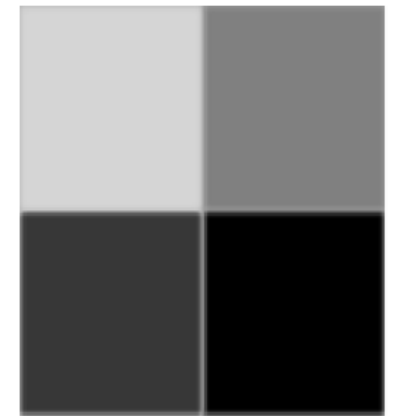
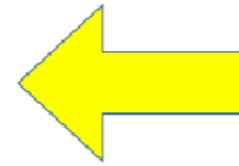
Unity

- 타일링(Tiling)

Texture를
1:1로
매핑하였을 때



3D 평면에
적용한 모습

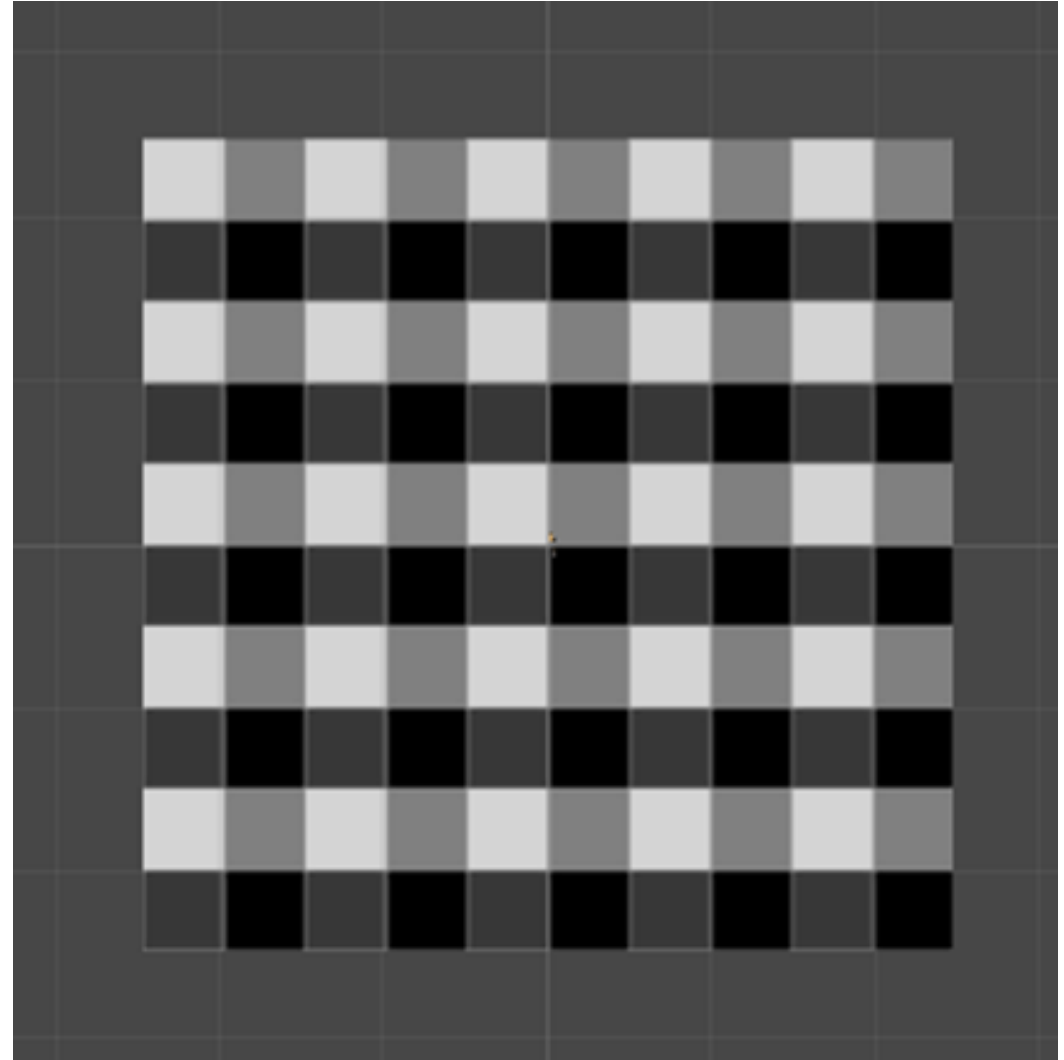


원본 이미지

Unity

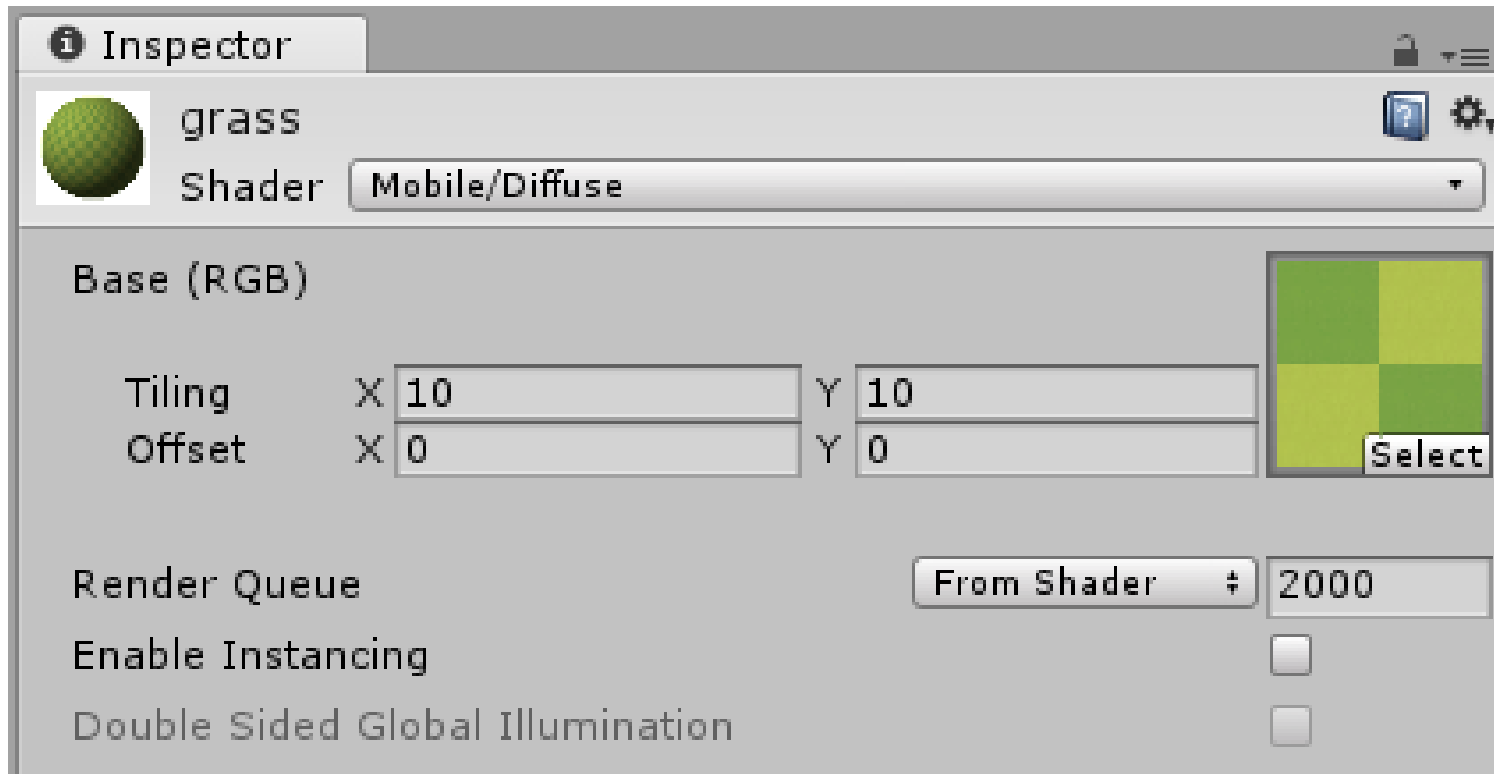
- 타일링 (Tiling)

반복적으로 그렸을 때



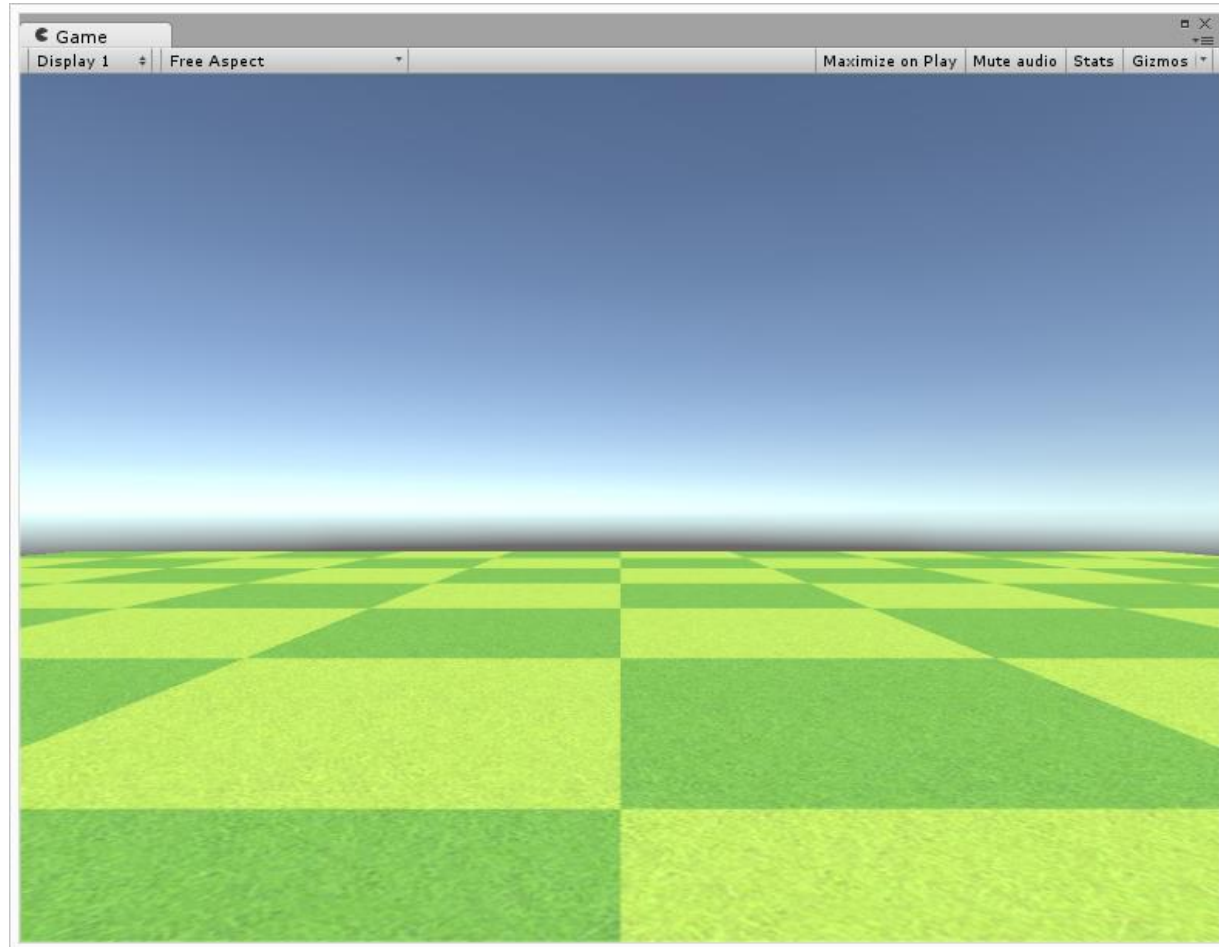
Unity

- 타일링 적용



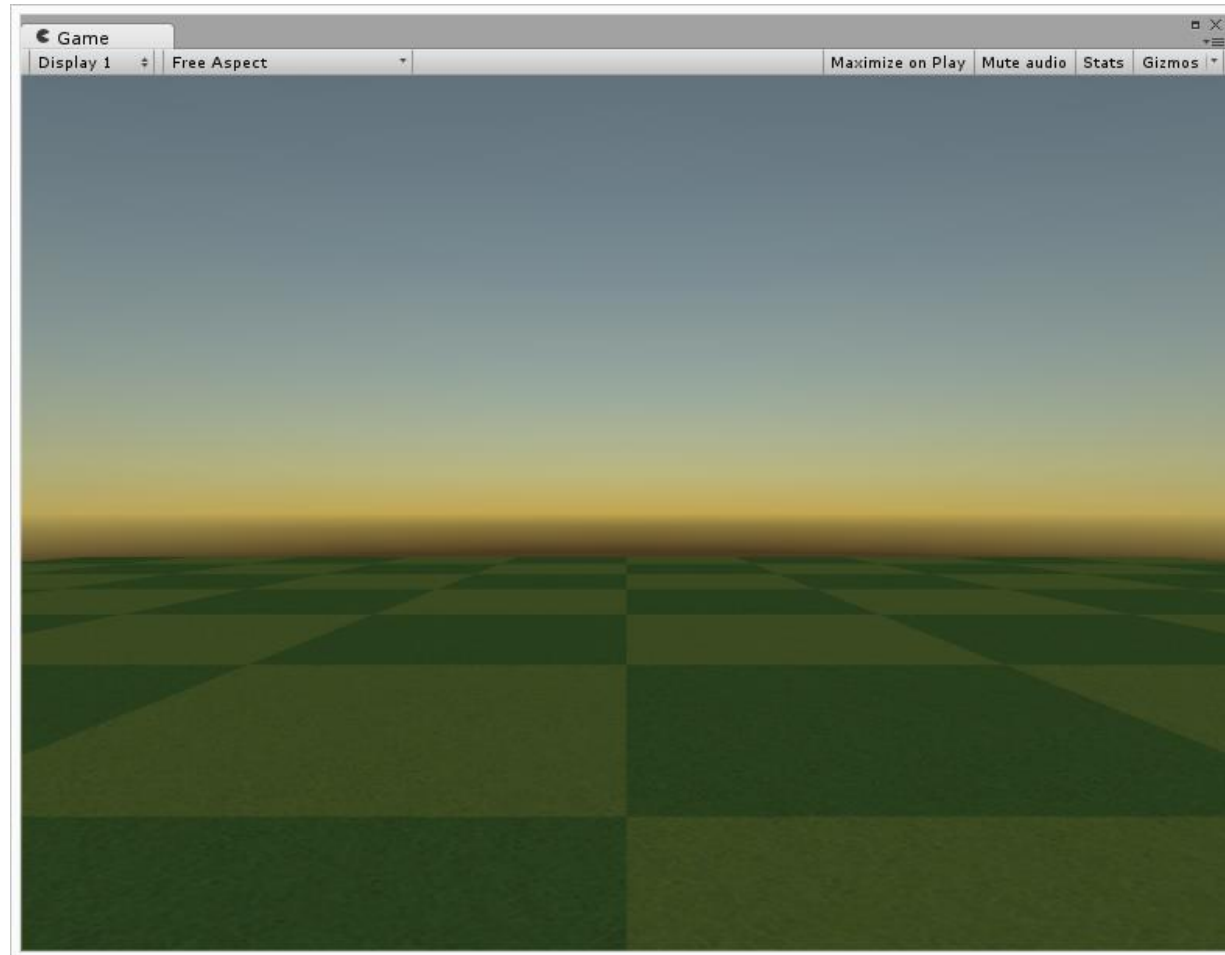
Unity

- 게임 뷰 확인



Unity

- 조명 제거 : 조명 변경을 위해 Directional Light 제거



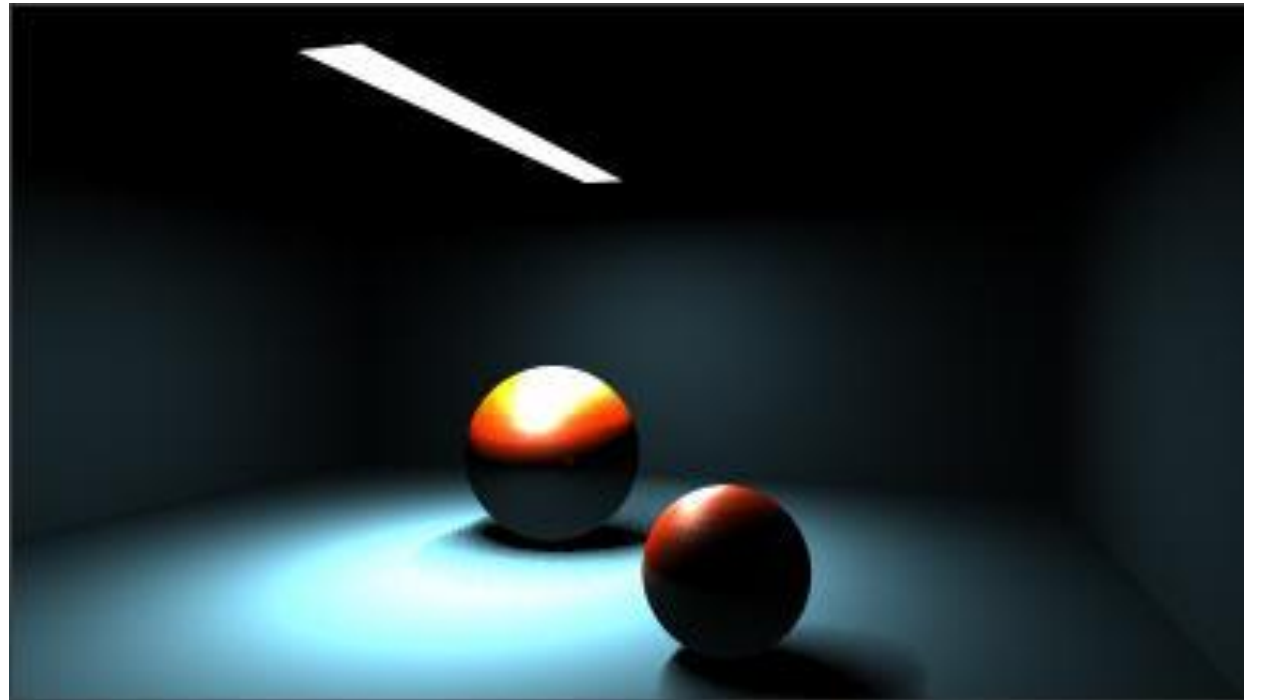
Unity – 조명

- 유니티 제공 조명
 - 1) Directional Light
 - 2) Point Light
 - 3) Spot Light
 - 4) Area Light
(Only for lightmap)
 - 5) Ambient Light



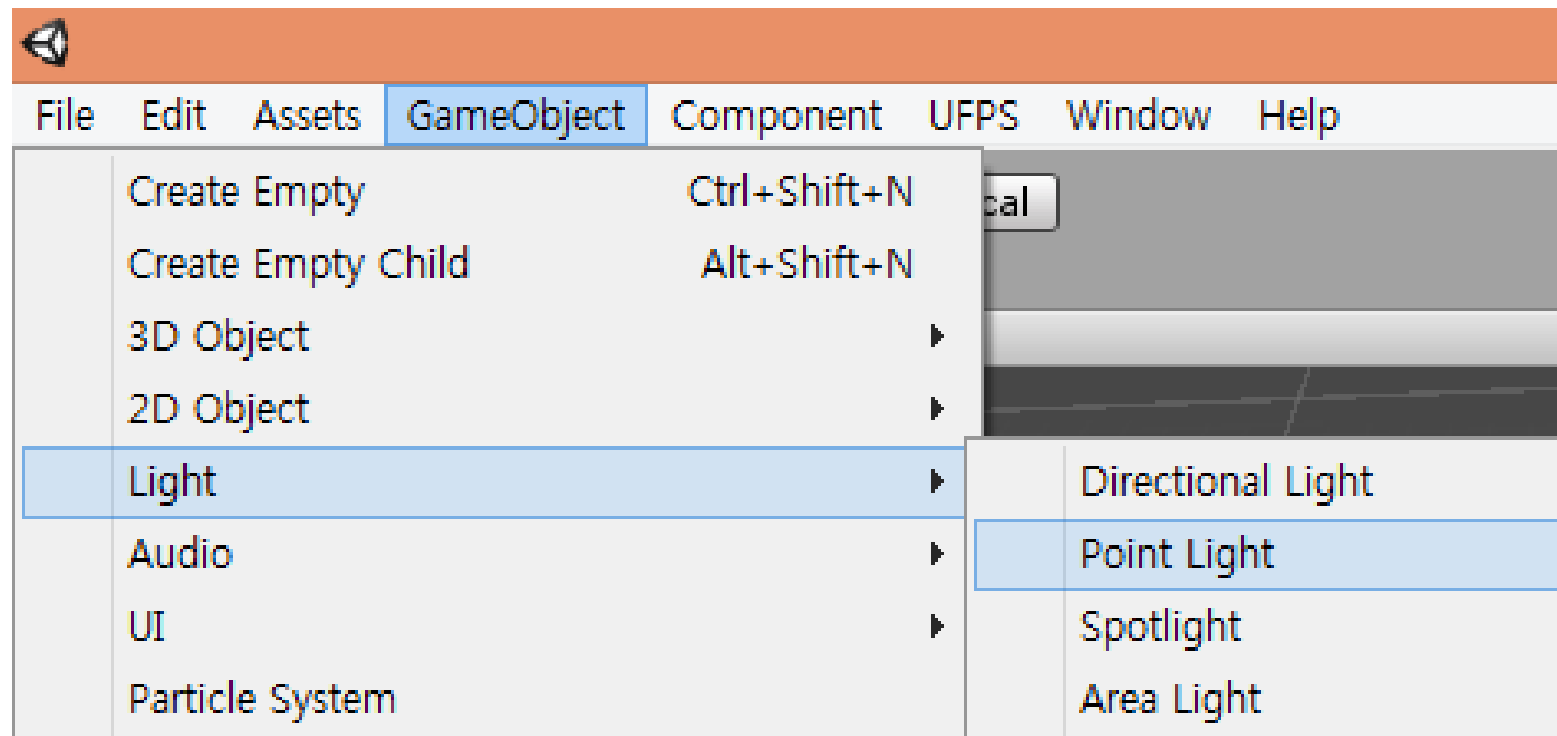
Unity – 조명

- 유니티 제공 조명
 - 1) Directional Light
 - 2) Point Light
 - 3) Spot Light
 - 4) Area Light
(Only for lightmap)
 - 5) Ambient Light



Unity

- 조명 설치 : Point light 추가



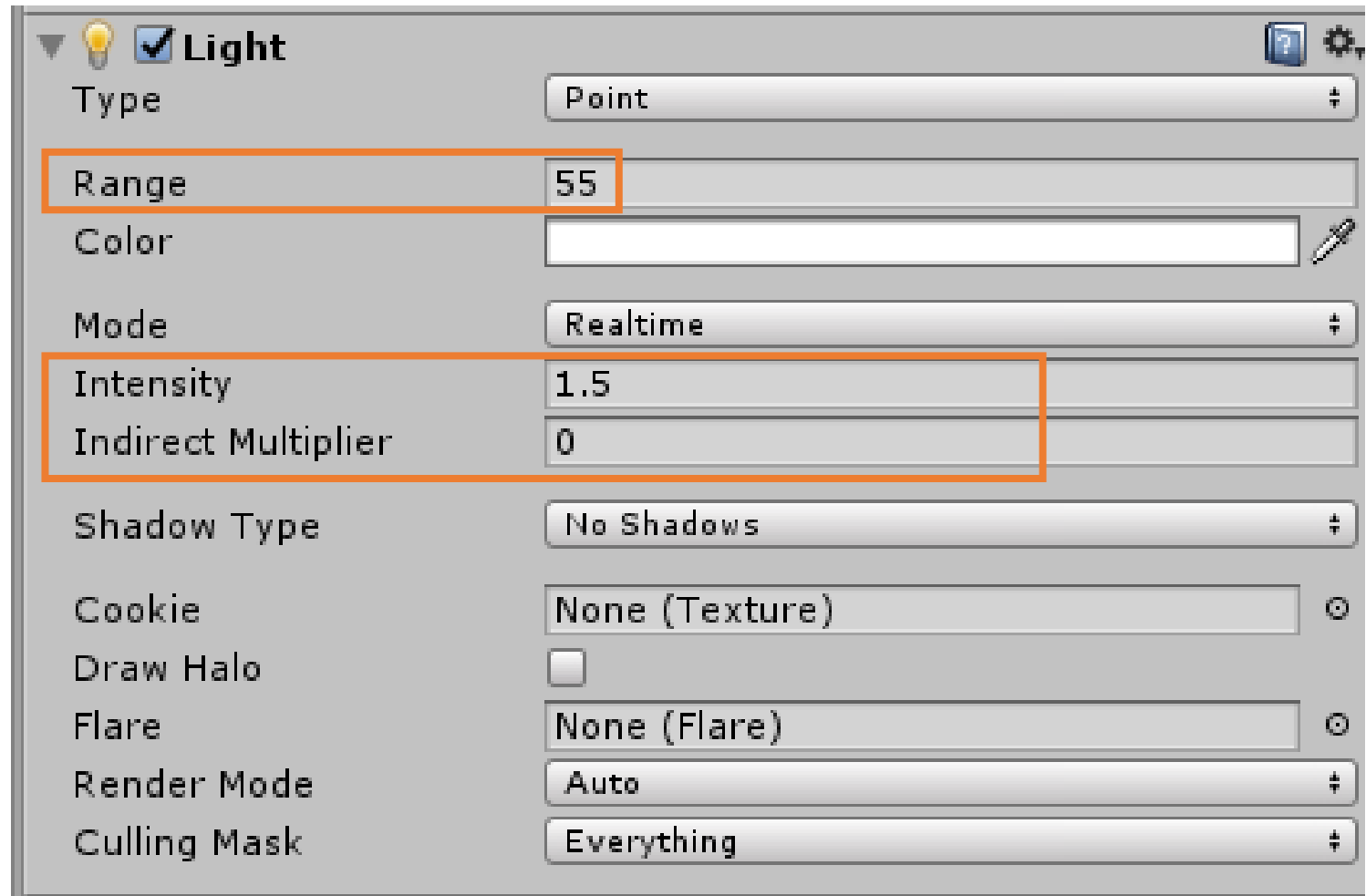
Unity

- 조명 설치
: Point light 추가
y = 10

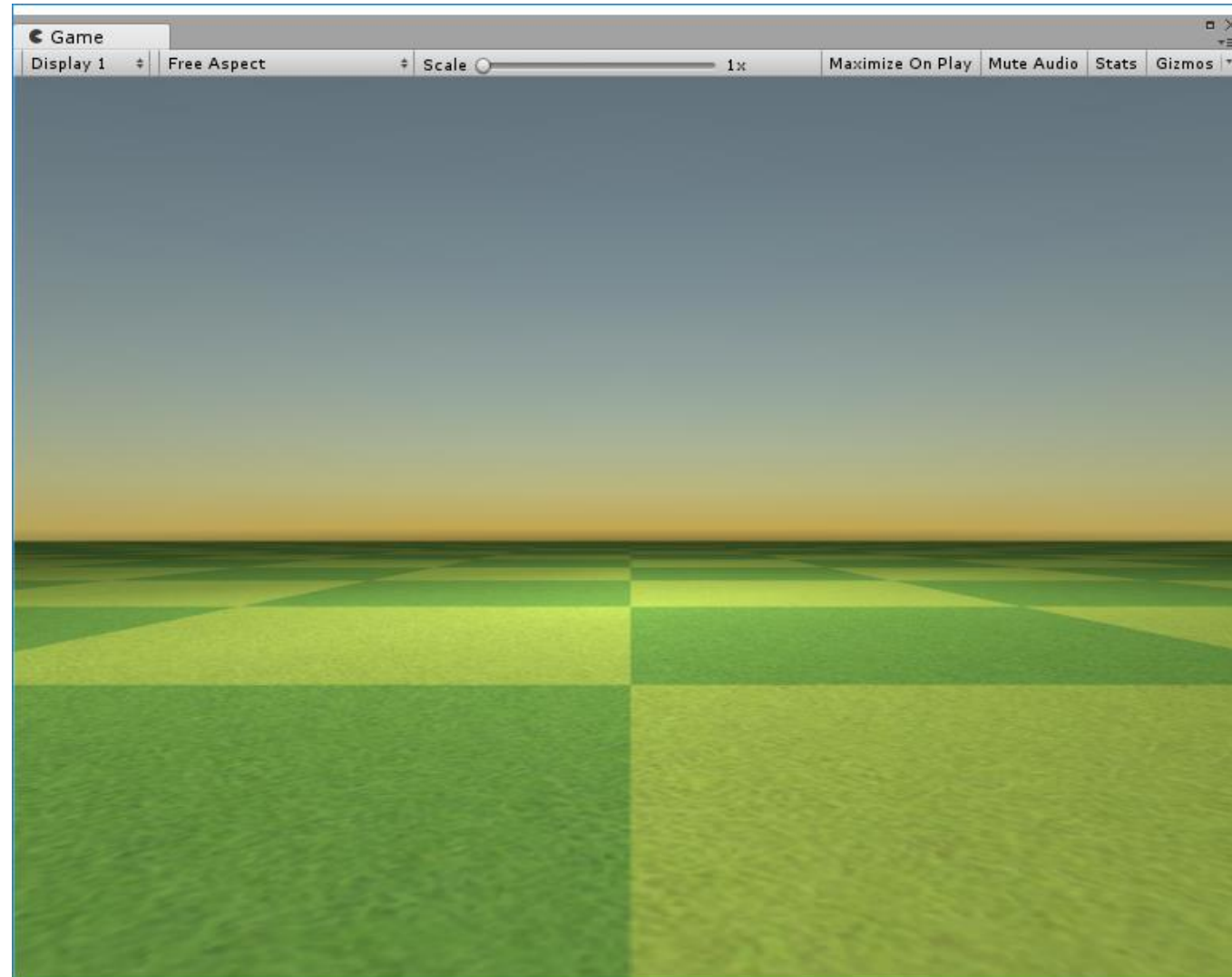
range = 55

Intensity = 1.5

Indirect Multiplier = 0



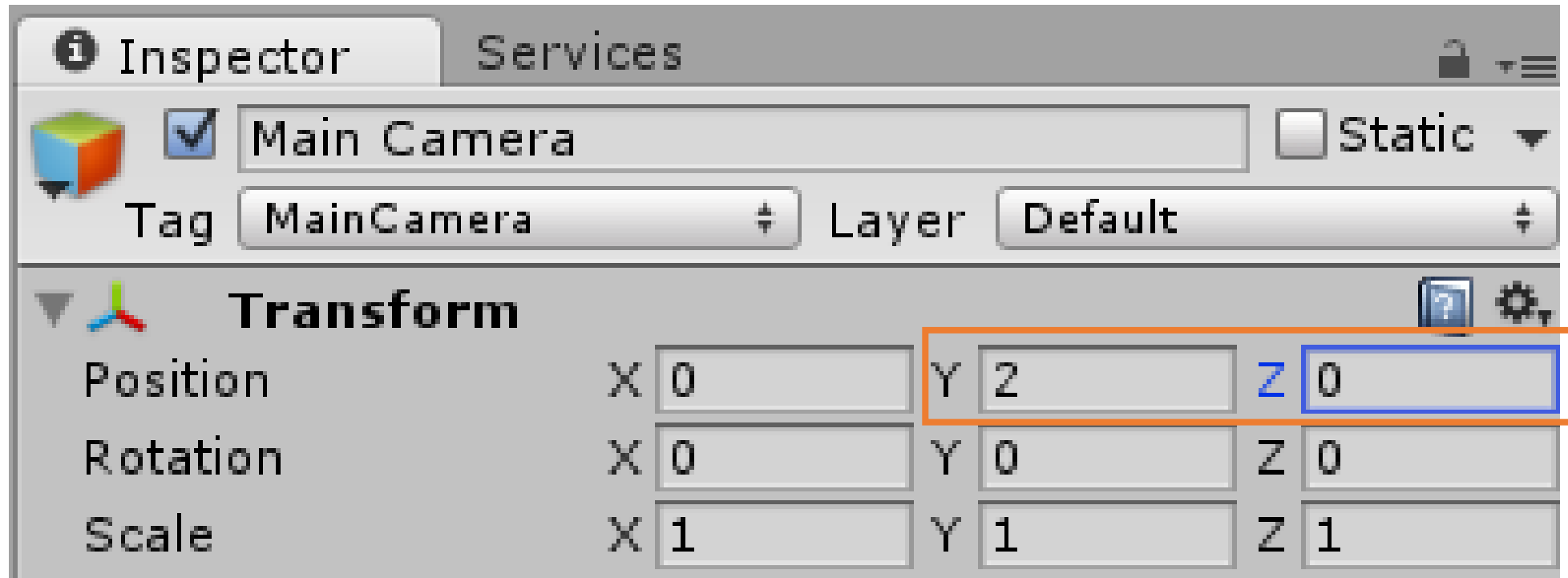
Unity



Unity

- 카메라 위치 조절

y : 2 z : 0



Unity – 컴퍼넌트 구현

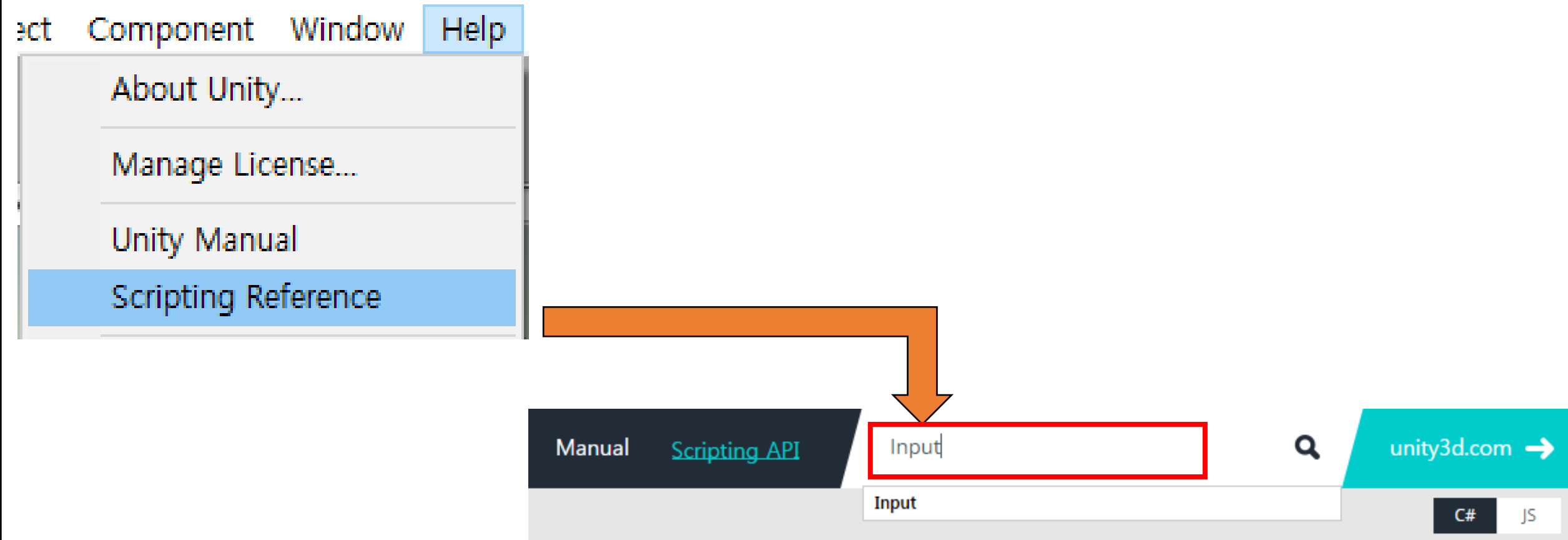
- 컴퍼넌트 구현

CameraControl

```
void Update()
{
    float  mouseMoveValueX  =  Input.GetAxis( "Mouse X" );
    float  mouseMoveValueY  =  Input.GetAxis( "Mouse Y" );
}
```


Unity – 컴퍼넌트 구현


- Input 클래스 내용 살펴보기



Unity – 컴퍼넌트 구현

- Input 클래스 내용 살펴보기

 **unity** | DOCUMENTATION

Manual [Scripting API](#) Input 

Version: 2017.1b ([switch to 5.6](#)) C# JS

Scripting API

- + UnityEngine
- + UnityEditor
- + Other

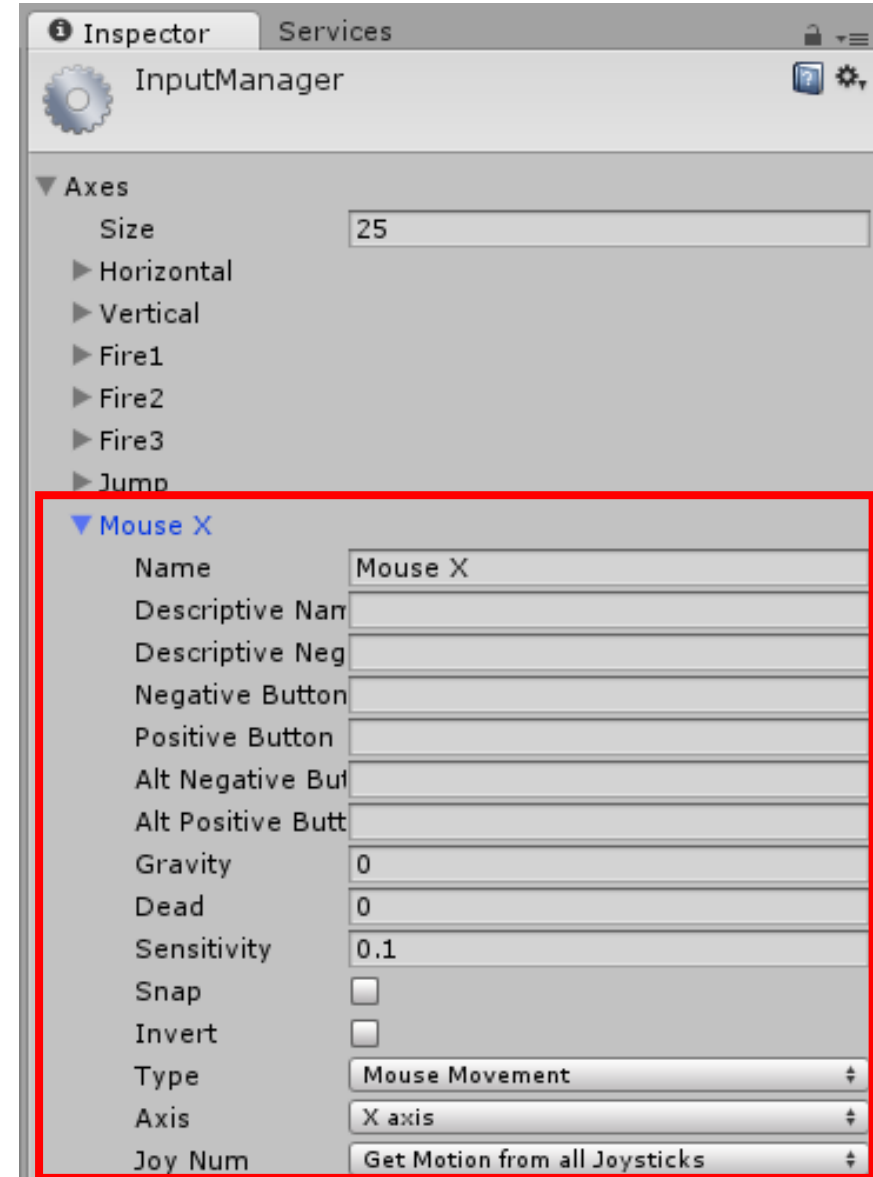
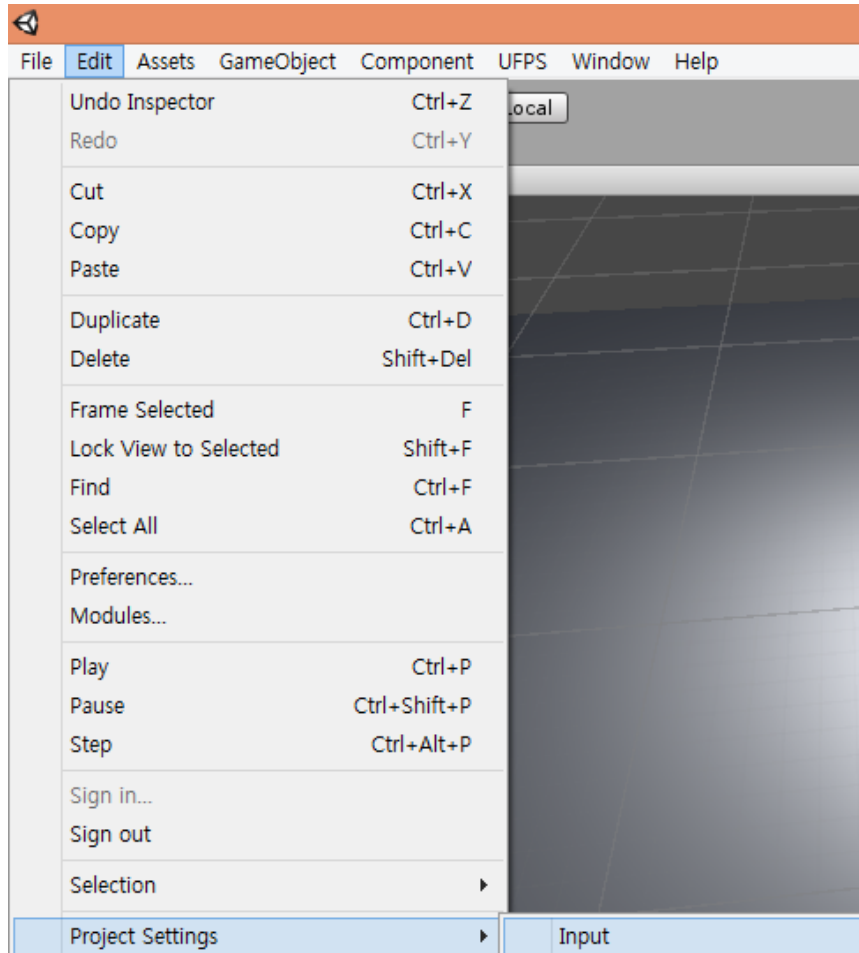
Your search for "**Input**" resulted in 429 matches:

[Input](#)
Interface into the Input system.

[InputAction](#)

Unity – 컴퍼넌트 구현

- Mouse X, Mouse Y 확인



```
public class CameraControl : MonoBehaviour
{
    public float sensitivity = 700.0f;
    float rotationX;
    float rotationY;

    void Update()
    {
        float mouseMoveValueX = Input.GetAxis("Mouse X");
        float mouseMoveValueY = Input.GetAxis("Mouse Y");

        rotationY += mouseMoveValueX * sensitivity * Time.deltaTime;
        rotationX += mouseMoveValueY * sensitivity * Time.deltaTime;

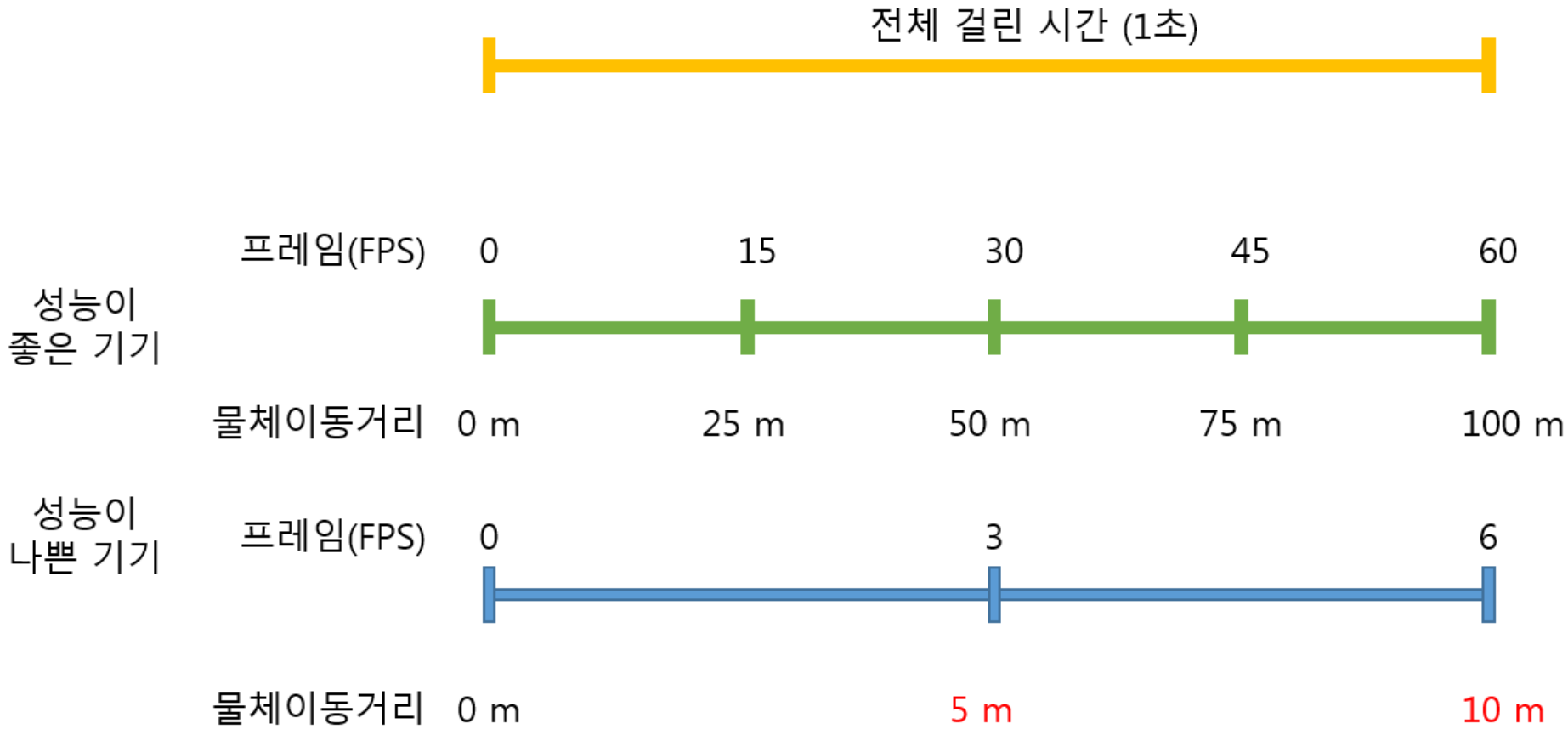
        rotationX %= 360;
        rotationY %= 360;

        transform.eulerAngles = new Vector3( -rotationX, rotationY, 0.0f );
    }
}
```

Unity – 컴퍼넌트 구현

- deltaTime의 의미
: 이전 프레임과 현재 프레임 사이의 간격
- 시간을 곱하는 의미
: 기기 성능과 무관하게 항상 같은 결과를 만들어 내기 위해서

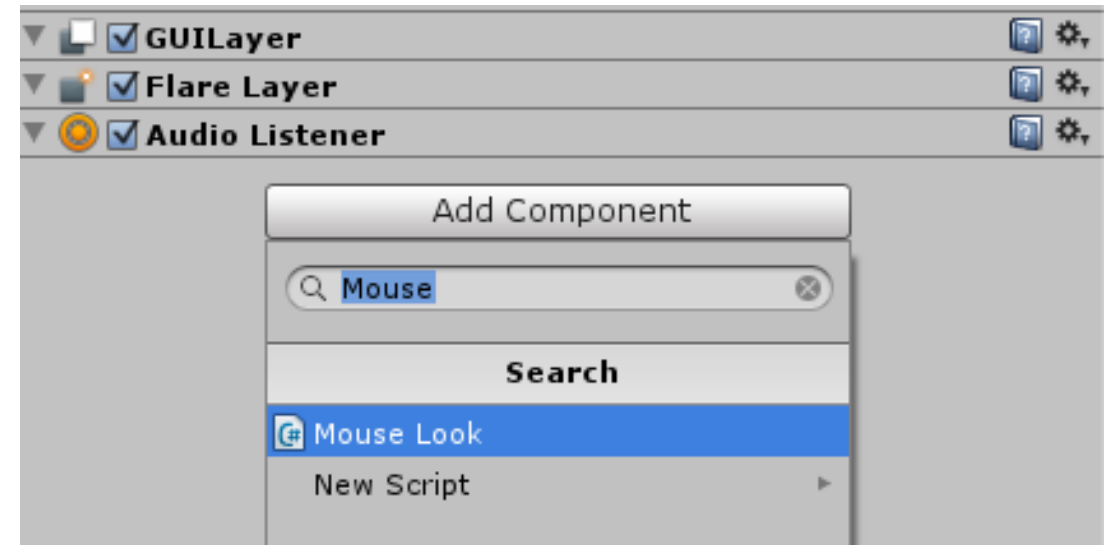
Unity – 컴퍼넌트 구현



Unity – 컴퍼넌트 구현

- 스크립트 연동

- 1) 스크립트 선택 후 계층 뷰 오브젝트에 연결
- 2) 추가할 오브젝트 선택 후
스크립트를 클릭하고 드래그 하여 인스펙터에 연결
- 3) 추가할 오브젝트 선택 후
[**Add Component**] 버튼 클릭 후
메뉴에서 추가



Unity – 컴퍼넌트 구현

- Play시 카메라 제약이 없으므로 제약 사항을 추가해야 한다

[Hint]

1. Debug.Log 를 활용하여 [콘솔(Console)]창에서 변수의 값을 변화를 알아본다.
2. 값의 변화를 통해 각도 제한을 구현