UNITY

Intermediate

Custom Editor

Custom Editor

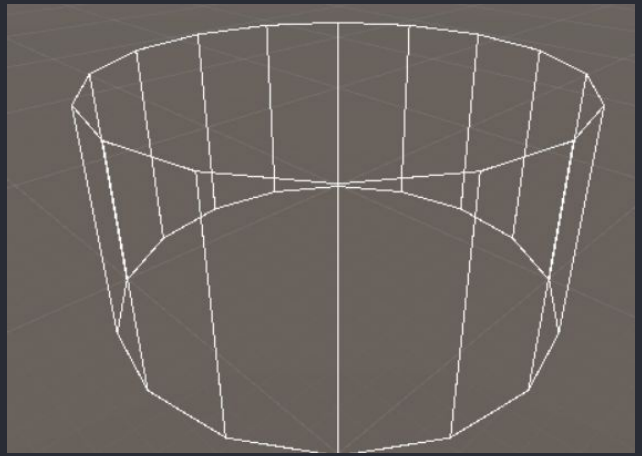
유니티에서 개발을 할 때, 에디터를 커스텀해야할 필요가 있을 경우가 존재합니다. 예를 들어, 특정 좌표들을 순회하는 Waypoint 기능이 필요할 때, 빈 GameObject들을 일일이 배치해서 등록하는 것 보다는 버튼을 누를 때마다 Waypoint 오브젝트가 생성되고, 이 Waypoint마다 선으로 연결해서 실제 동선을 보여주면 더 좋겠죠.

또는, 범위내에 있는 플레이어를 감지하고 추적하려 할때, 범위를 구 형태로 보여주고, 시야각을 부채꼴로 보여준 후에 유닛이 보고 있는 시점을 초록색 선으로 표현하다가 플레이어가 감지되면 빨간색으로 변하는 등의 디버깅이 필요할 수도 있겠죠. 이들을 가능하게 해주는 기능들이 유니티에 존재합니다.

OnDrawGizmos()

먼저, 가장 간단한 Gizmos 기능입니다. 이 기능은 MonoBehavior에 존재하고 있으며, Scene 화면에 그려주기 위해 존재합니다.
OnDrawGizmos() 는 Scene에 계속 표시해주고,
OnDrawGizmosSelected() 는 선택되었을 때만 Scene에 표시해줍니다.

그럼 이제 Gizmos의 속성과 메서드들을 알아보도록 하죠.



Gizmos.color

Gizmos의 속성은 총 4가지가 존재하며, 전부 static입니다. 이 중에서 color라는 속성 말고는 거의 쓰이지 않으며, color 속성은 중요하기도 합니다.

```
Gizmos.color = Color.red
```

이 속성을 변경하여 표시되는 색상을 변경할 수 있습니다. Color 클래스를 받습니다.

Draw Methods

Gizmos에는 여러 도형을 그리는 static 메서드들을 지원하며, 이중 자주 쓰이는 것들을 설명해 드리겠습니다. 만약 모든 메서드를 보고 싶으시다면 다음 링크를 참조하여주세요. 사용법은 간단합니다

Unity - Scripting API: Gizmos (unity3d.com)

Line, Cube, Sphere

`Gizmos.DrawLine(Vector3 from, Vector3 to)`
from 좌표부터 to 좌표까지 선을 그립니다

`Gizmos.DrawCube(Vector3 center, Vector3 size)`
center 좌표에 size 만큼의 큐브를 그립니다

`Gizmos.DrawSphere(Vector3 center, Vector3 radius)`
center 좌표에 radius 반지름 만큼의 구체를 그립니다

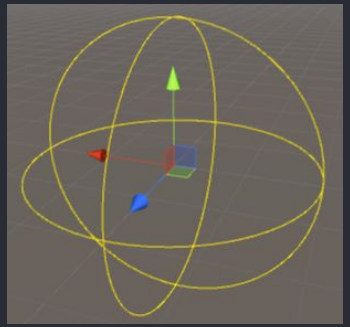
예시로 아래와 같은 모양으로 생성됩니다. 이 외에도 Texture를 이용해 그리거나, Mesh를 그릴수도 있습니다.

WireCube, WireSphere

Gizmos.DrawWireCube(Vector3 center, Vector3 size)
center 좌표에 size 만큼의 와이어 큐브를 그립니다

Gizmos.DrawWireSphere(Vector3 center, Vector3 radius)
center 좌표에 radius 반지름 만큼의 와이어 구체를 그립니다

와이어란, 아래와 같이 윤곽선만 표시되는 것을 말합니다.



ExecuteInEditMode

```
[ExecuteInEditMode]
Unity 스크립트|참조 0개
public class EditorLookAt : MonoBehaviour
{
    public Transform target;

    Unity 메시지|참조 0개
    void Update()
    {
        transform.LookAt(target);
    }
}
```

위의 코드는 가장 빠르게 에디터 전용 기능을 만들 수 있는데, 클래스 위에 [ExecuteInEditMode] 를 넣음으로써 에디터 내에서 실행되게 할 수 있습니다. 위의 스크립트를 아무 GameObject에나 등록한 후에, target에 원하는 GameObject를 등록해주면 해당 게임오브젝트를 에디터 환경에서도 계속해서 바라본다는 것을 알 수 있죠!

Custom Editor 제작

방금 코드는 에디터에서 실행하게 해주지만, Inspector 창의 구조를 바꿀 수는 없습니다. 그래서 특정 컴포넌트가 Inspector에서 나타내는 방식을 변경하는 것이 에디터 클래스(Editor Class) 입니다.

이는 강의로 설명하기에는 많은 내용이 있어 공식 메뉴얼을 참고하시는 것을 추천드립니다.

Unity - Manual: Custom Editors (unity3d.com)

```
using UnityEngine;
using UnityEditor;

[CustomEditor(typeof(LookAtPoint))]
[CanEditMultipleObjects]
참조 0개
public class LookAtPointEditor : Editor
{
    SerializedProperty lookAtPoint;

    참조 0개
    void OnEnable()
    {
        lookAtPoint = serializedObject.FindProperty("lookAtPoint");
    }

    참조 0개
    public override void OnInspectorGUI()
    {
        serializedObject.Update();
        EditorGUILayout.PropertyField(lookAtPoint);
        serializedObject.ApplyModifiedProperties();
    }
}
```