编写一个简单的自定义 Component，用自定义组件定义几种飞碟，做成预制

•参考官方手册 https://docs.unity3d.com/ScriptReference/Editor.html

•实现自定义组件，编辑并赋予飞碟一些属性

Transform组件变量

属性	说明
position	在世界坐标系中,transform的位置
localPosition	相对于父级的变换的位置
eulerAngles	旋转作为欧拉角度
localEulerAngles	相对于父级的变换的旋转欧拉角度
rotation	在世界坐标系中物体变换的旋转角度作为Quaternion储存
parent	返回物体变换的父级
root	返回最高层次的

为一个组件添加自己自定义的编辑器内容

```
[CustomEditor(typeof(ShowArea))]
```

Editor.OnInspectorGUI

通过实现该函数来制作自定义的Inspector面板。

```
0 Unity 消息 | 0 个引用
private void OnGUI()
{
    // GUILayout.Width  控制在窗口中物体所在的宽
    // GUILayout.Height  控制在窗口中物体所在的高
    //它们的返回的类型为GUILayoutOption
    GUILayout.Label 提示语句
    GUILayout.Button( 按钮
    GUILayout.TextField 文本
    GUILayout.Space 空行
    EditorGUILayout.Toggle 开关（跟Toggle一样）
    GUILayout.BeginHorizontal 横向
    GUILayout.BeginVertical 纵向
    GUILayout.HorizontalSlider（横） GUILayout.VerticalSlider（纵）  Slider(分横纵 上横下纵)
    EditorGUILayout.Popup  下拉
    GUILayout.BeginScrollView  滑动列表
    EditorGUILayout.BoundsField （边界输入）  EditorGUILayout.ColorField（颜色输入）  EditorGUILayout.CurveField（曲线输入）  输入框
    EditorGUILayout.TagField   tag(标签)
    EditorGUILayout.LayerField(可以获取所有的Layer)
    EditorGUILayout.MaskField (下拉可以多选)
    EditorGUILayout.ObjectField(选择物体)
    EditorGUILayout.Foldout 折叠
    EditorGUILayout.BeginToggleGroup(开关)
    GUILayout.FlexibleSpace（布局之间左右对齐）
    EditorGUILayout.HelpBox（提示语句）
    EditorGUILayout.Slider(Slider)
    EditorGUILayout.TextArea（text 自适应高）
    GUILayout.PasswordField(可以改变成对应的符号)
    EditorGUILayout.Vector2Field  EditorGUILayout.Vector3Field  EditorGUILayout.Vector4Field
    EditorGUILayout.SelectableLabel （可以复制粘贴）
    EditorGUILayout.MinMaxSlider （取值范围）
    EditorGUILayout.IntSlider（只能是整数）
    EditorGUILayout.InspectorTitlebar（将物体返回回来）
}
}
```

CSDN @一只学习的小鸟

```
    Vector3 startPosition = EditorGUILayout.Vector3Field("StartPosition",UFO.startPosition); //文本，数值域
    UFO.startPosition = startPosition;
```

结果展示

```
//UFO.cs
using UnityEngine;

namespace HitUFO
{
    public class UFO : MonoBehaviour
    {
        public int score = 0; //设置飞碟得分
        public static Vector3 startPosition = new Vector3(0, 0, 0); //设置初始位置
        public static Vector3 startSpeed = new Vector3(1, 1, 0); //设置初始速度
        public static Vector3 localScale = new Vector3(1, 1, 1); //设置缩放比例
        private int Left_or_Right;

        public Vector3 GetSpeed()
        {
            //向左还是向右运动
            Vector3 v = startSpeed;
            v.x *= Left_or_Right;
            return v;
        }

        public void SetSide(int lr,float dy)
        {
            Vector3 v = startPosition;
            v.x *= lr;
            v.y += dy;
            transform.position = v;
            Left_or_Right = lr;
        }

        public void SetLocalScale(float x,float y,float z)
        {
            Vector3 lc = localScale;
            lc.x *= x;
            lc.y *= y;
            lc.z *= z;
            transform.localScale = lc;
        }
    }
}
```

```
//SetGui.cs
using UnityEngine;
using UnityEditor;

namespace HitUFO{
```

```
[CustomEditor(typeof(UFO))]
public class SetGui : Editor //继承editor类
{
    public override void OnInspectorGUI()
    {
        var target = (UFO)serializedObject.targetObject; //获取对象

        EditorGUILayout.Space(); //空行
        Vector3 startPosition = EditorGUILayout.Vector3Field("StartPosition",UFO.startPosition); //文本，数值域
        UFO.startPosition = startPosition;

        EditorGUILayout.Space();
        Vector3 startSpeed = EditorGUILayout.Vector3Field("StartSpeed",UFO.startSpeed);
        UFO.startSpeed = startSpeed;

        EditorGUILayout.Space();
        Vector3 localScale = EditorGUILayout.Vector3Field("LocalScale",UFO.localScale);
        UFO.localScale = startPosition;
    }
}
}
```