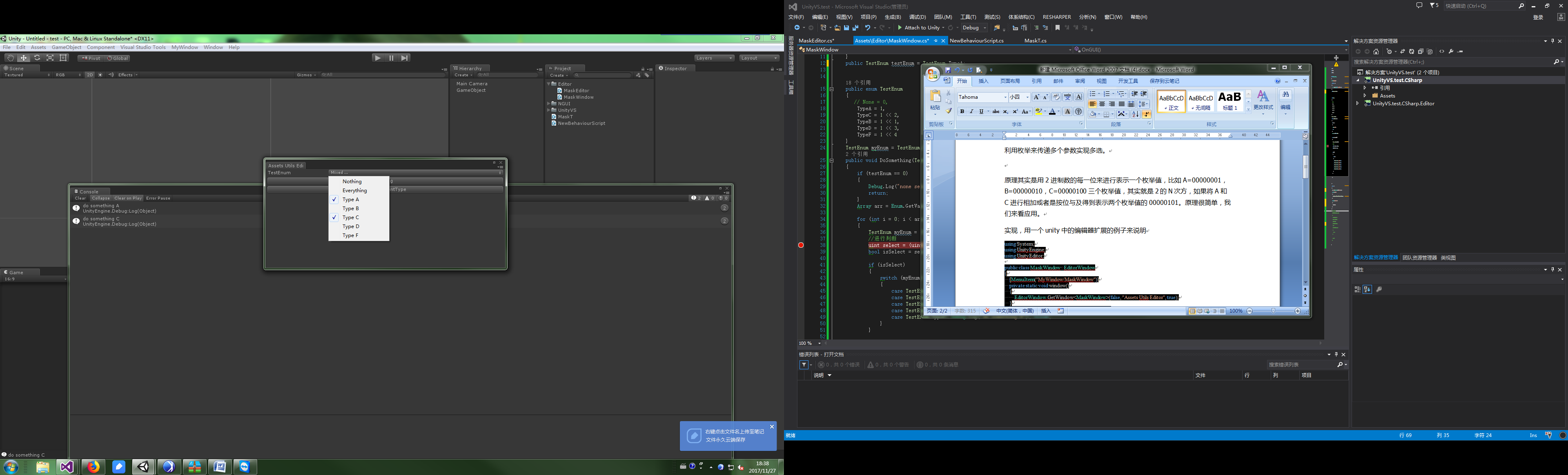
**枚举多选的原理与实现**

# 利用枚举来传递多个参数实现多选。



原理其实是用2进制数的每一位来进行表示一个枚举值，比如A=00000001，B=00000010，C=00000100三个枚举值，其实就是2的N次方，如果将A和C进行相加或者是按位与及得到表示两个枚举值的00000101，这个代表多个值的枚举值，我们将其简称为多重枚举（此名字为杜撰）。原理很简单，我们来看应用。

实现，用一个unity中的编辑器扩展的例子来说明

using System;

using UnityEngine;

using UnityEditor;

public class MaskWindow : EditorWindow

{

[MenuItem("MyWindow/MaskWindow")]

private static void window()

{

EditorWindow.GetWindow<MaskWindow>(false, "Assets Utils Editor", true);

}

public TestEnum testEnum = TestEnum.TypeA;

public enum TestEnum

{

// None = 0,

TypeA = 1,

TypeC = 1 << 2,

TypeB = 1 << 1,

TypeD = 1 << 3,

TypeF = 1 << 4

}

TestEnum myEnum = TestEnum.TypeA;

public void DoSomething(TestEnum testEnum)

{

if (testEnum == 0)

{

Debug.Log("none select");

return;

}

Array arr = Enum.GetValues(typeof(TestEnum)); //这个会得到一个按值大小排名的枚举数组，如我上面TypeB和TypeC在代码上的位置调换。

for (int i = 0; i < arr.Length; i++)

{

TestEnum myEnum = (TestEnum)arr.GetValue(i);

//进行判断

uint select = (uint)myEnum & (uint)testEnum;

bool isSelect = select == (uint)myEnum;

if (isSelect)

{

switch (myEnum)

{

case TestEnum.TypeA: Debug.Log("do something A"); break;

case TestEnum.TypeB: Debug.Log("do something B"); break;

case TestEnum.TypeC: Debug.Log("do something C"); break;

case TestEnum.TypeD: Debug.Log("do something D "); break;

case TestEnum.TypeF: Debug.Log("do something F"); break;

}

}

}

}

private void OnGUI()

{

testEnum = (TestEnum)EditorGUILayout.EnumMaskField("TestEnum", testEnum);

if (GUILayout.Button("debug"))

{

DoSomething(TestEnum.TypeA | TestEnum.TypeB | TestEnum.TypeC | TestEnum.TypeD);

}

if (GUILayout.Button("Debug testEnum"))

{

DoSomething(testEnum);

}

}

}

当然枚举在定义的时候也可以用常用的0,1,2,3,4自然数来进行赋值如：

public enum TestEnum

{

// None = 0,

TypeA = 0,

TypeC,//1 << 2,

TypeB, //1 << 1,

TypeD,//1 << 3,

TypeF,//1 << 4

}

，只需要在判断的时候进行手动移位，将上面判断语句改为

TestEnum myEnum = (TestEnum)arr.GetValue(i);

//进行判断

int myEnumint = 1 << i;

uint select = (uint)myEnumint & (uint)testEnum;

bool isSelect = select == (uint)myEnumint;

if (isSelect)

当然在传递“多重枚举”值的时候也没有上面方便了，同样需要进行位移

int temp = 1 << (int)TestEnum.TypeA | 1 << (int)(TestEnum.TypeD);

DoSomething((TestEnum)temp);

当然由于其原理的制约导致32位的uint只能表示32个枚举值。