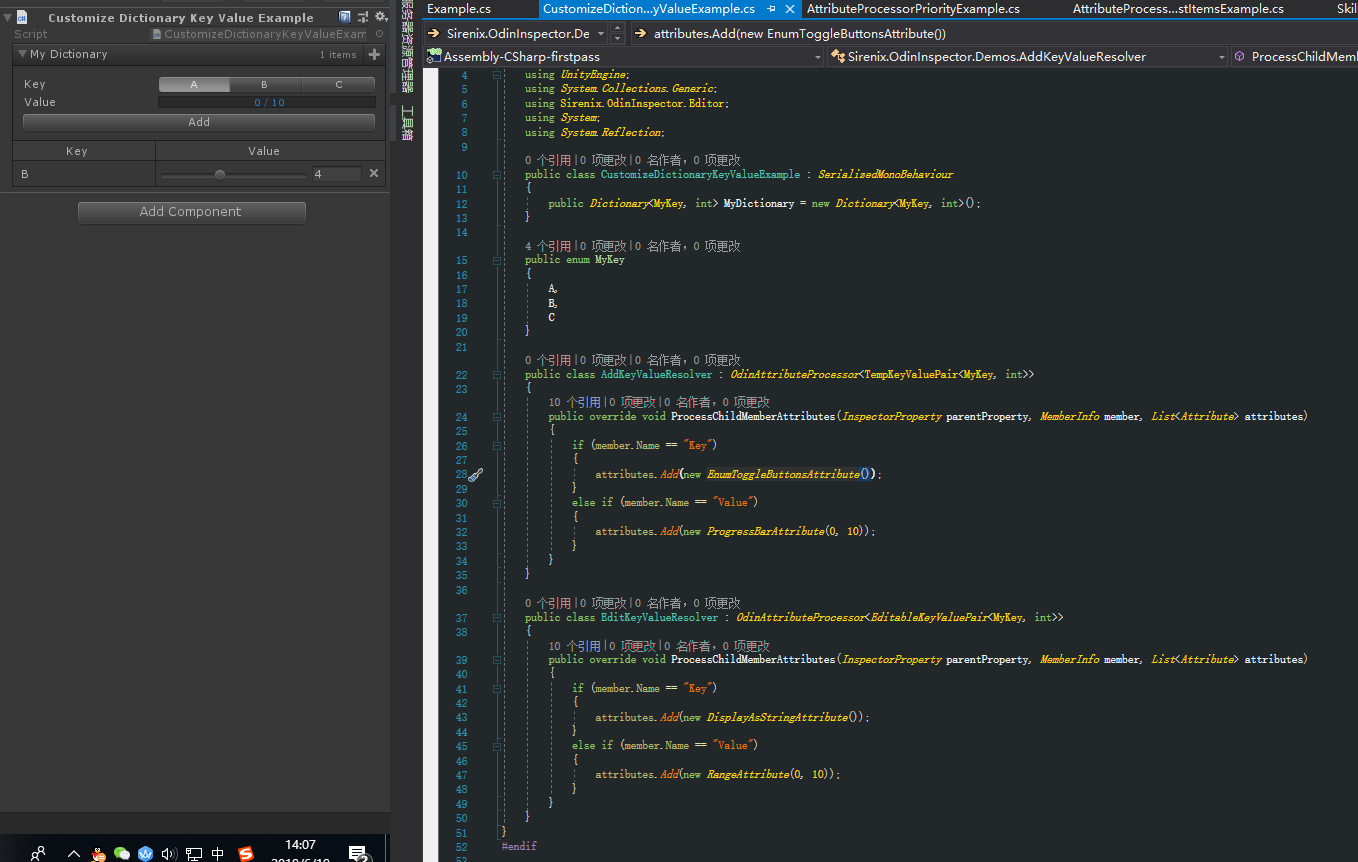
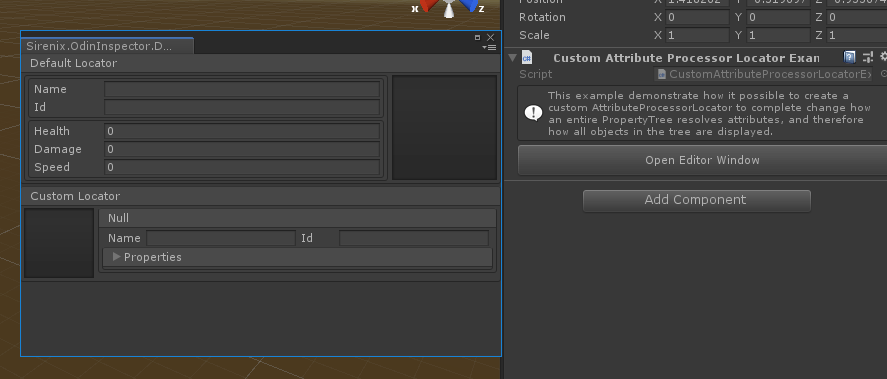
添加字典结构



打开编辑面板，添加自定义编辑面板



#if UNITY\_EDITOR

namespace Sirenix.OdinInspector.Demos

{

using *System*;

using *System*.*Collections*.*Generic*;

using *System*.*Reflection*;

using Sirenix.OdinInspector.Editor;

using UnityEngine;

using *UnityEditor*;

using Sirenix.Utilities.*Editor*;

using Sirenix.Utilities;

[TypeInfoBox("This example demonstrate how it possible to create a custom AttributeProcessorLocator to complete change how an entire PropertyTree resolves attributes, and therefore how all objects in the tree are displayed.")]

public class CustomAttributeProcessorLocatorExample : *MonoBehaviour*

{

[Button(*ButtonSizes*.*Large*)]

private void OpenEditorWindow()

{

var window = Editor.*CreateInstance*<SomeCustomEditorWindow>();

window.*Show*();

window.*position* = *GUIHelper*.*GetEditorWindowRect*().*AlignCenter*(500, 300);

}

}

public class SomeCustomEditorWindow : *UnityEditor*.*EditorWindow*

{

private *PropertyTree* defaultPropertyTree;

private *PropertyTree* customPropertyTree;

private void OnEnable()

{

this.*wantsMouseMove* = true;

}

private void OnGUI()

{

this.DrawWithDefaultLocator(); //

this.DrawWithCustomLocator(); //绘制CustomLocator

this.*RepaintIfRequested*();

}

private void DrawWithDefaultLocator()

{

if (this.defaultPropertyTree == null)

{

this.defaultPropertyTree = *PropertyTree*.*Create*(new SomeClass());

}

*SirenixEditorGUI*.*BeginBox*("Default Locator");

this.defaultPropertyTree.*Draw*(false);

*SirenixEditorGUI*.*EndBox*();

}

private void DrawWithCustomLocator()

{

if (this.customPropertyTree == null)

{

this.customPropertyTree = *PropertyTree*.*Create*(new SomeClass());

this.customPropertyTree.*AttributeProcessorLocator* = new CustomMinionAttributeProcessorLocator();

}

*SirenixEditorGUI*.*BeginBox*("Custom Locator");

this.customPropertyTree.*Draw*(false);

*SirenixEditorGUI*.*EndBox*(); //顶到最后

}

}

public class SomeClass

{

[HorizontalGroup("Split", *LabelWidth* = 80)]

[BoxGroup("Split/$Name", showLabel: false)]

[BoxGroup("Split/$Name/NameId", showLabel: false)]

public string Name, Id;

[HideLabel, PropertyOrder(5)]

[PreviewField(*Height* = 105), HorizontalGroup("Split", *width*: 105)]

public *Texture2D* Icon;

[BoxGroup("Split/$Name/Properties", showLabel: false)]

public int Health, Damage, Speed;

}

[OdinDontRegister] // This attributes prevents Odin from using this AttributeProcessor in the default attribute resolver locator.

public class CustomMinionAttributeProcessor : *OdinAttributeProcessor*<SomeClass>

{

public override void ProcessChildMemberAttributes(*InspectorProperty* parentProperty, *MemberInfo* member, *List*<*Attribute*> attributes)

{

attributes.*Clear*(); // Get rid of all other attributes.

switch (member.*Name*)

{

case "Icon":

attributes.*Add*(new *HorizontalGroupAttribute*("Split", *width*: 70));

attributes.*Add*(new *PreviewFieldAttribute*(70, OdinInspector.*ObjectFieldAlignment*.*Left*));

attributes.*Add*(new *PropertyOrderAttribute*(-5));

attributes.*Add*(new *HideLabelAttribute*());

break;

case "Name":

case "Id":

attributes.*Add*(new *BoxGroupAttribute*("Split/$Name"));

attributes.*Add*(new *VerticalGroupAttribute*("Split/$Name/Vertical"));

attributes.*Add*(new *HorizontalGroupAttribute*("Split/$Name/Vertical/NameId"));

attributes.*Add*(new *LabelWidthAttribute*(40));

break;

default:

attributes.*Add*(new *FoldoutGroupAttribute*("Split/$Name/Vertical/Properties", *expanded*: false));

attributes.*Add*(new *LabelWidthAttribute*(60));

break;

}

}

}

public class CustomMinionAttributeProcessorLocator : *OdinAttributeProcessorLocator*

{

private static readonly CustomMinionAttributeProcessor Processor = new CustomMinionAttributeProcessor();

public override *List*<*OdinAttributeProcessor*> GetChildProcessors(*InspectorProperty* parentProperty, *MemberInfo* member)

{

return new *List*<*OdinAttributeProcessor*>() { Processor };

}

public override *List*<*OdinAttributeProcessor*> GetSelfProcessors(*InspectorProperty* property)

{

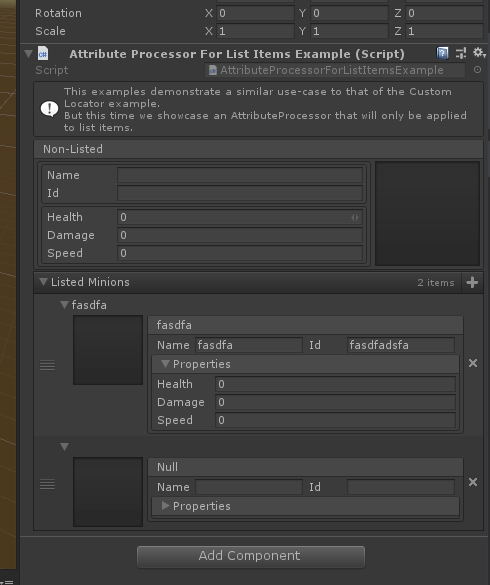
return new *List*<*OdinAttributeProcessor*>() { Processor };

}

}

}

#endif



表格结构

