

vInspector Manual

Attributes

- Button attribute

- Size attribute

- Space attribute

- Foldout attribute

- Tab attribute

- Variants attribute

Resettable variables

Cleaner header

Script asset inspector

Disabling parts of vInspector

Attributes

Attributes allow you to create and group UI elements in inspector without writing custom editors

Add this line to your script to use attributes:

```
using VInspector;
```

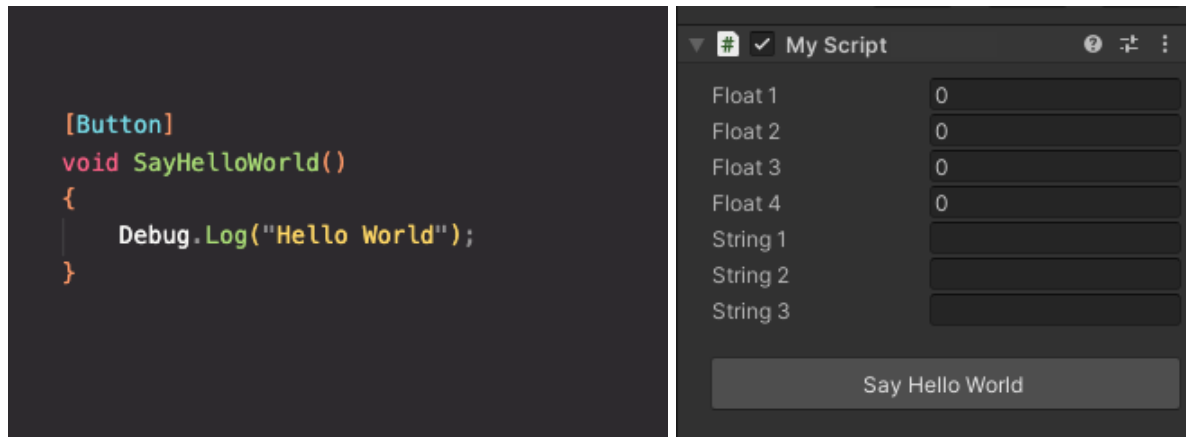
If you want attributes to retain their state after recompilation (e.g. foldouts staying folded or expanded), add this variable to your script:

```
public VInspectorData vInspectorData;
```

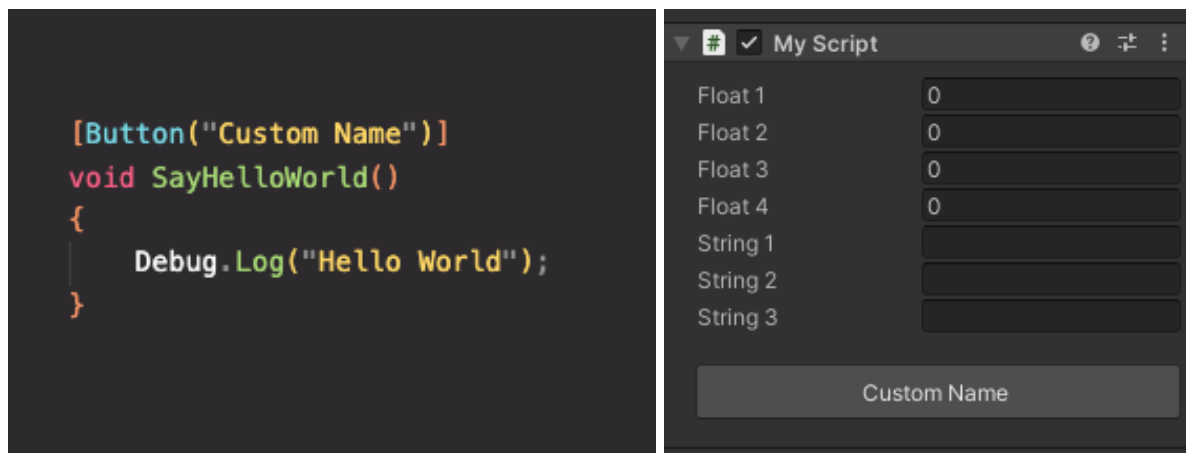
Button attribute

Creates a button at the bottom of inspector

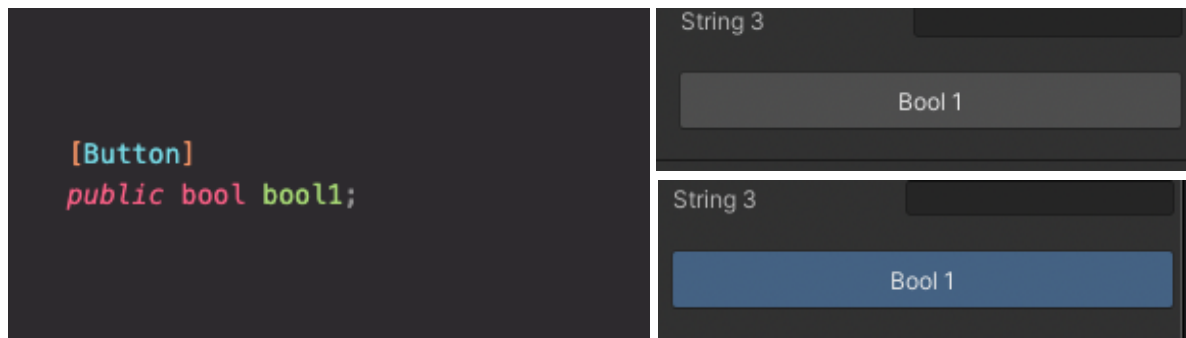
Add it before the function you want the button to invoke:



You can assign a custom name to the button:



Buttons can be used to toggle bools (button appears pressed when bool is true):



Size attribute

Use it to change button size:

```
[Button]
[Size(40)]
void BigButton()
{

}

[Button]
[Size(22)]
void SmallButton()
{

}
```

▼ # ✓ My Script ? ↕ ⋮

Float 1	0
Float 2	0
Float 3	0
Float 4	0
String 1	
String 2	
String 3	

Big Button

Small Button

Space attribute

Use it to add space between variables or buttons:

```
public float float1;
public float float2;
public float float3;
public float float4;

[Space]
public string string1;
public string string2;
public string string3;

[Button]
void FirstButton()
{

}

[Space]
[Button]
void SecondButton()
{

}
```

My Script

Float 10

Float 20

Float 30

Float 40

String 1

String 2

String 3

First Button

Second Button

Add Component

Add this line to your script to enable it:

```
using Space = VInspector.SpaceAttribute;
```

You can change the amount of space by passing it as argument:

```
[Space(100)]
public string string1;
public string string2;
public string string3;
```

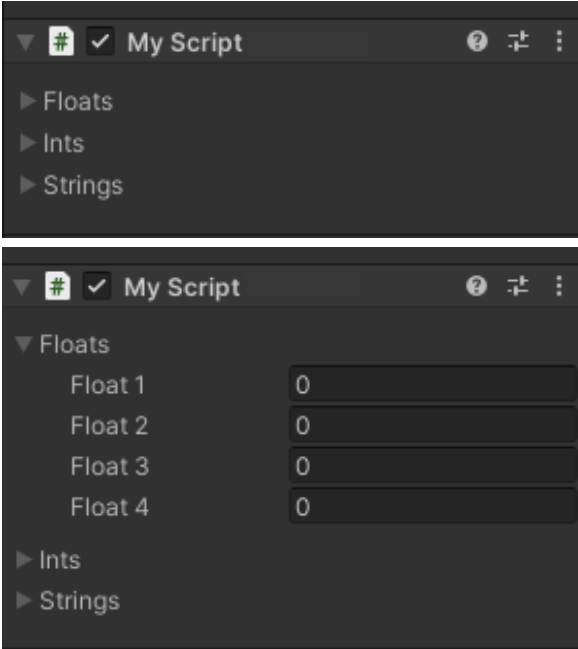
Foldout attribute

Use it to group variables into foldouts:

```
[Foldout("Floats")]
public float float1;
public float float2;
public float float3;
public float float4;

[Foldout("Ints")]
public int int1;
public int int2;
public int int3;

[Foldout("Strings")]
public string string1;
public string string2;
public string string3;
```



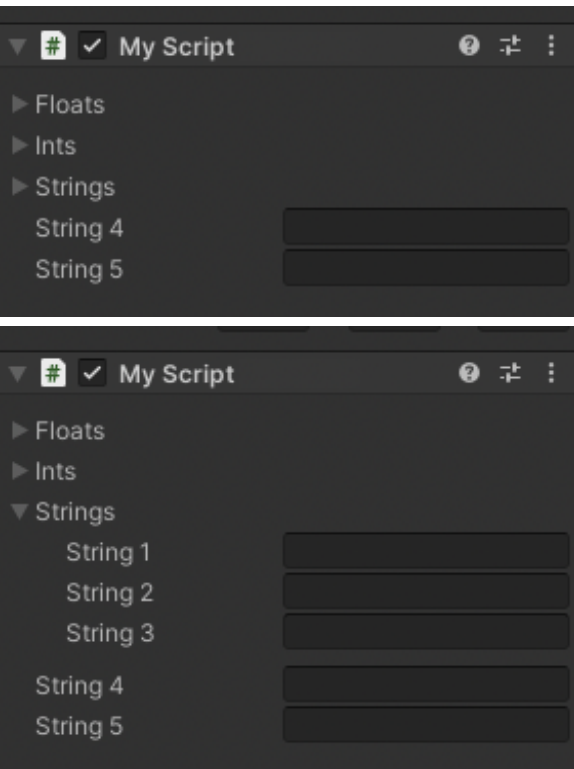
Use EndFoldout attribute to prevent grouping variables into foldout:

```
[Foldout("Floats")]
public float float1;
public float float2;
public float float3;
public float float4;

[Foldout("Ints")]
public int int1;
public int int2;
public int int3;

[Foldout("Strings")]
public string string1;
public string string2;
public string string3;
[EndFoldout]

public string string4;
public string string5;
```



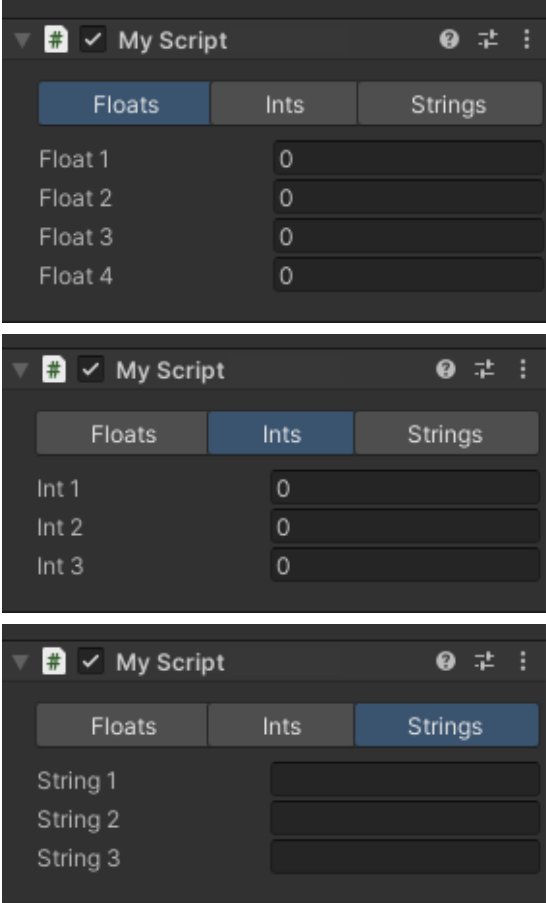
Tab attribute

Use it to group variables or buttons into tabs:

```
[Tab("Floats")]
public float float1;
public float float2;
public float float3;
public float float4;

[Tab("Ints")]
public int int1;
public int int2;
public int int3;

[Tab("Strings")]
public string string1;
public string string2;
public string string3;
```

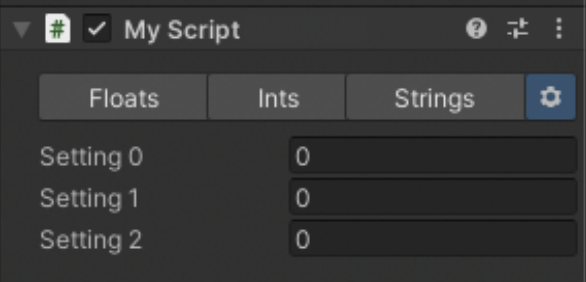


The image displays three sequential screenshots of a script editor window titled "My Script". Each screenshot shows a different tab selected from a set of three tabs: "Floats", "Ints", and "Strings".

- Floats Tab:** The "Floats" tab is selected. It contains four input fields labeled "Float 1", "Float 2", "Float 3", and "Float 4", each with the value "0".
- Ints Tab:** The "Ints" tab is selected. It contains three input fields labeled "Int 1", "Int 2", and "Int 3", each with the value "0".
- Strings Tab:** The "Strings" tab is selected. It contains three input fields labeled "String 1", "String 2", and "String 3", each with an empty text box.

If the last tab is called Settings, it will appear as an icon instead of text:

```
[Tab("Settings")]
public int setting0;
public int setting1;
public int setting2;
```



The image shows a screenshot of a script editor window titled "My Script". The "Settings" tab is selected, which is represented by a gear icon instead of text. It contains three input fields labeled "Setting 0", "Setting 1", and "Setting 2", each with the value "0".

Like with foldouts, you can prevent grouping into tabs by using EndTab attribute

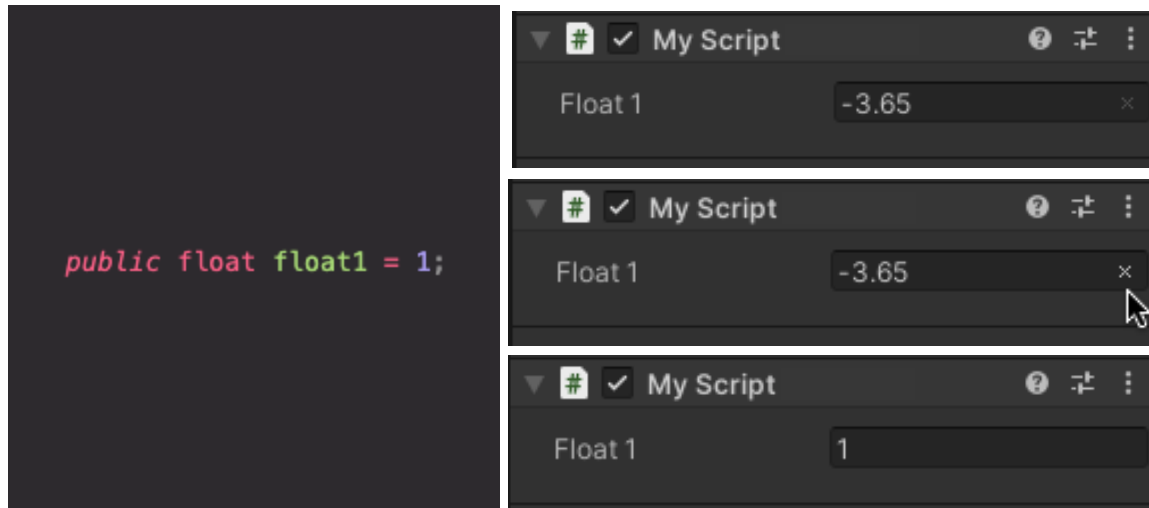
Variants attribute

Use it to create a dropdown for setting strings to predefined variants:



Resettable variables

Variables can be resetted to default value by clicking on the cross button:



If the script is attached to a prefab instance, the value on the original prefab is considered default, otherwise default is the value you defined in script

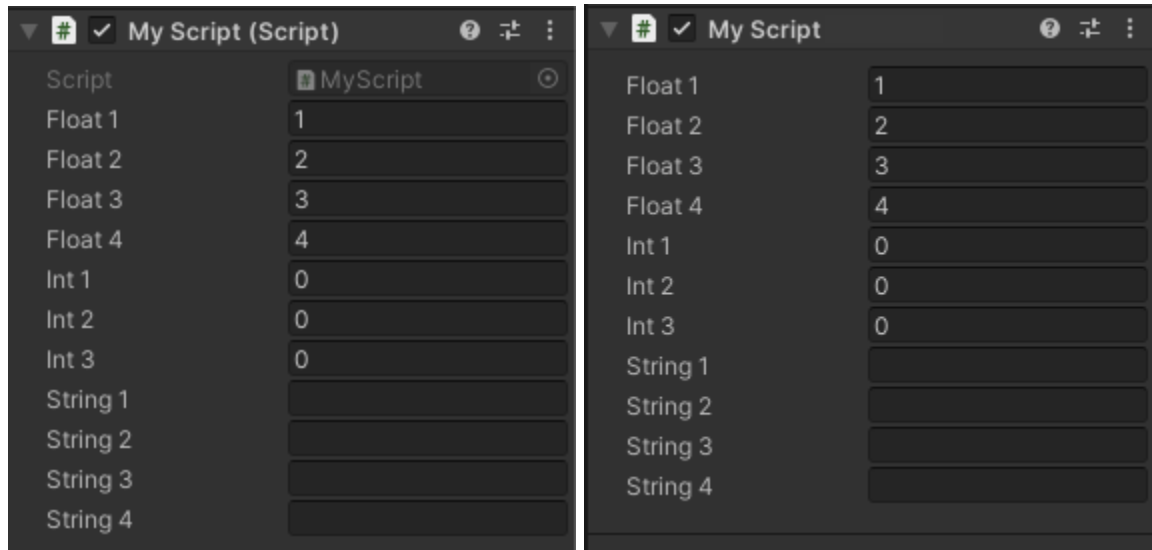
This feature works out of the box, no code needed

Except if you want it work with unity's Range attribute, add this line:

```
using Range = VInspector.RangeAttribute;
```

Cleaner header

vInspector hides the greyed-out Script field and “(Script)” text at the top:

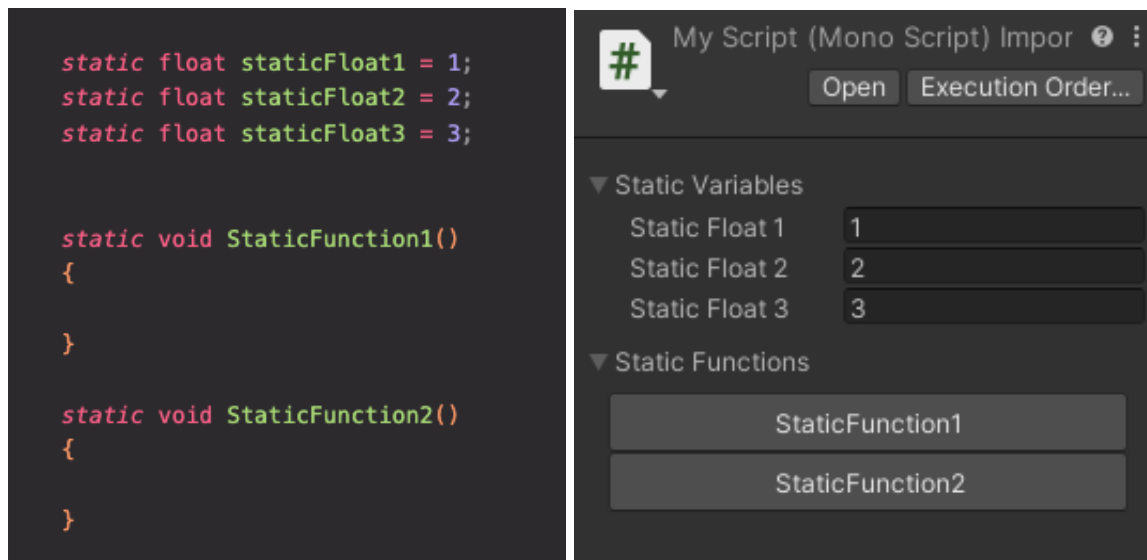


Also it allows you to open the script by double-clicking the script name or to show the script in project browser by alt-clicking the name

These features work out of the box, no code needed

Script asset inspector

Inspector allows you to see static variables and invoke static functions from script asset inspector:

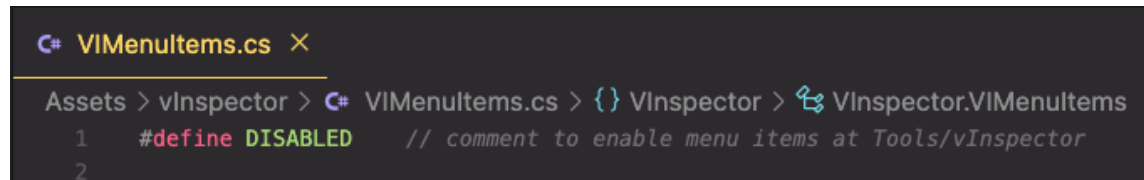


Useful when you want to run some code without creating a GameObject and attaching a script to it

This feature works out of the box, no code needed

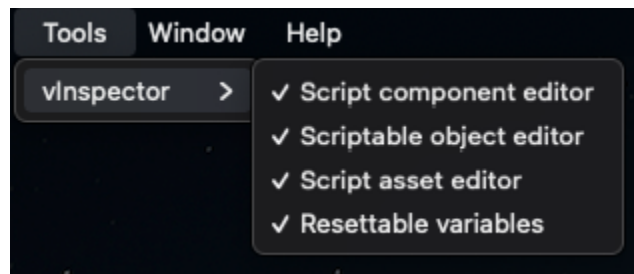
Disabling parts of vInspector

If you want to disable some aspects of vInspector, open VIMenuItems.cs and comment the first line:



```
C# VIMenuItems.cs X
Assets > vInspector > C# VIMenuItems.cs > {} VInspector > VInspector.VIMenuItems
1  #define DISABLED    // comment to enable menu items at Tools/vInspector
2
```

This will enable menu items at Tools/vInspector:



May be useful if vInspector interferes with your custom editors or other plugins, which is unlikely

If it does happen to be the case or if you have any questions, please contact us kubacho.labs@gmail.com