

Reorderable Arrays

a Unity3D Editor Extension

USER GUIDE

Release 13th of October 2016

Table of Contents

Table of Contents	2
Welcome	3
Contact Info	3
Overview	3
Quick Start	3
Features Overview	4
Reorder Array Elements	4
Add and Remove Array Elements	4
Folding the Array	4
Custom Editors	5
Scripting Reference	6
Class: ReorderableArrayEditor	6
ReorderableArrayEditor.useReorderableArrays	6
ReorderableArrayEditor.useNestedReorderableArrays	6
ReorderableArrayEditor.DrawDefaultInspector	7
ReorderableArrayEditor.DrawProperty	7
ReorderableArrayEditor.DrawReorderableArray	7
ReorderableArrayEditor.HasNotReorderableAttribute	7
Attribute: NotReorderableAttribute	8

Welcome

Thank you for purchasing Reorderable Arrays! This editor extension will improve your workflow with arrays in the inspector. It works out of the box and can even be used with custom inspectors. As soon as you drop the ReorderableArray.dll into your project, all inspector arrays will upgrade. I hope this editor extension will save you time and improve your workflow. If you like it, please support me by a rating on the Asset Store. If you have any questions please contact me!

Contact Info

For support, bugs and feature requests, you can send an email to: support@tiny-plugin.com (english and german)

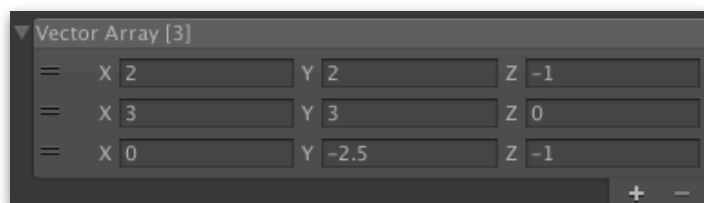
Overview

This guide was designed to provide Reorderable Arrays users a basic overview of the editor extension.

Quick Start

Use the Demo scene to see the ReorderableArray in action. Follow these easy steps:

1. Load the "Demo" scene
2. Click on the "ArrayTest" scene object, to see it in the inspector
3. See the reorderable arrays in the inspector
4. The last array called "floatArray" has the `[NotReorderable]` attribute attached and is displayed with the default unity inspector for arrays



Features Overview

- works out of the box, just add the ReorderableArray.dll into your project
- reorder array elements by dragging and dropping into a new position
- add new array elements by pressing the '+' button
- delete array elements by selecting them and pressing the '-' button
- fold and unfold the whole array by pressing the '▼' button
- drag and drop support: new array elements can be dragged to the Reorderable Array in the inspector

Reorder Array Elements

The main feature in Reorderable Arrays is the reorder feature. To use it, you simple grab an array element by its handle (at the left side) and move it by dragging it up or down to the desired position.



Add and Remove Array Elements

An other important feature is the ability to add new elements to the array as well as to remove unwanted elements from the array. To accomplish this tasks there are two buttons at the bottom right of the Reorderable Array.

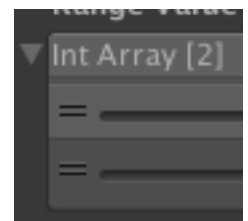


Use the '+' button to add new elements at the last position of the array.

Use the '-' button to remote selected array elements. The '-' will be disabled if no element is selected.

Folding the Array

In order to gain overview in the inspector you are able to collapse Reorderable Arrays like the default arrays in the inspector. Just use the '▼' button near to the Reorderable Arrays.



Custom Editors

If your project contains custom Editors that use following attribute, they will interfere with ReorderableArrays. This will disable reorderable arrays and fall back to default arrays.

```
[CustomEditor(typeof(MonoBehaviour), true, isFallback = true)]
```

To resolve this issue, you could simple extend from ReorderableArraysEditor instead of Editor.

```
using UnityEngine;
using UnityEditor;

[CustomEditor(typeof(MonoBehaviour), true, isFallback = true)]
[CanEditMultipleObjects]
public class CustomEditorTest : ReorderableArrayEditor {

    public override void OnInspectorGUI() {
        DrawDefaultInspector();
    }
}
```

Scripting Reference

To add the Reorderable Arrays to a custom editor just simply extend from `ReorderableArrayEditor` instead of `Editor`

Class: ReorderableArrayEditor

Namespace: `UnityEditor` Inherits from: `Editor`

DESCRIPTION

Custom Editor which enables the Reorderable Arrays

PROPERTIES

<code>useReorderableArrays</code>	Indicating if the editor should render the Reorderable Arrays
<code>useNestedReorderableArrays</code>	Indicating if the editor will render nested arrays also as Reorderable Arrays

FUNCTIONS

<code>DrawDefaultInspector</code>	Draws the default Inspector but using Reorderable Arrays
<code>DrawProperty</code>	Draws the property. Uses Reorderable Arrays if <code>drawReorderableArrays</code> is set to <code>true</code> .
<code>DrawReorderableArray</code>	Draws the Reorderable Array
<code>HasNotReorderableAttribute</code>	Has the property the <code>NotReorderable</code> attribute?

ReorderableArrayEditor.useReorderableArrays

```
bool useReorderableArrays;
```

DESCRIPTION

Indicating if the editor should render the Reorderable Arrays or use the default implementation of Unity3D. The default value is set to `true`.

ReorderableArrayEditor.useNestedReorderableArrays

```
bool useNestedReorderableArrays;
```

DESCRIPTION

Indicating if the editor will render nested arrays also as Reorderable Arrays. The default value is set to `false`.

ReorderableArrayEditor.DrawDefaultInspector

```
bool DrawDefaultInspector();
```

DESCRIPTION

Draws the default Inspector but using ReorderableArrays if useReorderableArrays is set to true.

ReorderableArrayEditor.DrawProperty

```
bool DrawProperty(SerializedProperty property, bool  
drawReorderableArrays);
```

DESCRIPTION

Draws the property. Uses a Reorderable Array if drawReorderableArrays is set to true.

ReorderableArrayEditor.DrawReorderableArray

```
void DrawReorderableArray(SerializedProperty property);
```

DESCRIPTION

Draws the ReorderableArray. property must be a SerializedProperty representing an array type.

ReorderableArrayEditor.HasNotReorderableAttribute

```
bool HasNotReorderableAttribute(SerializedProperty property);
```

DESCRIPTION

Returns true if the NotReorderable attribute was found for the given SerializedProperty.

Attribute: NotReorderableAttribute

Namespace: UnityEngine Inherits from: PropertyAttribute

DESCRIPTION

This attribute is used to disable the Reorderable Arrays for specific properties.

```
using UnityEngine;

public class Demo : MonoBehaviour {

    [Header("Classic Array Test")]
    [Range(1, 10)]
    [NotReorderable]
    public float[] floatArray;
}
```