

# MIT License (Open Source)

The MIT License (MIT)

Copyright (c) 2013-2015 Rotorz Limited

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

---

©2013-2015 Rotorz Limited. All rights reserved.

# Contribution Agreement

This project is licensed under the [MIT License](#). To be in the best position to enforce these licenses the copyright status of this project needs to be as simple as possible. To achieve this the following terms and conditions must be met:

- All contributed content (including but not limited to source code, text, image, videos, bug reports, suggestions, ideas, etc.) must be the contributors own work.
- The contributor disclaims all copyright and accepts that their contributed content will be released to the public domain.
- The act of submitting a contribution indicates that the contributor agrees with this agreement. This includes (but is not limited to) pull requests, issues, tickets, e-mails, newsgroups, blogs, forums, etc.

# Rotorz.ReorderableList Namespace

Contains classes for reorderable list control.

## ► Classes

Class	Description
 <a href="#">AddMenuClickedEventArgs</a>	Arguments which are passed to the <a href="#">AddMenuClickedEvent</a> .
 <a href="#">ElementAdderMenuBuilder</a>	Factory methods to create <a href="#">IElementAdderMenuBuilder</a> instances that can be used to build element adder menus.
 <a href="#">ElementAdderMenuCommandAttribute</a>	Annotate <a href="#">IElementAdderMenuBuilder</a> implementations with this attribute to associate it with addable elements.
 <a href="#">ElementAdderMeta</a>	Provides meta info useful when creating implementations of <a href="#">IElementAdderMenuBuilder</a> interface.
 <a href="#">GenericListAdapterT</a>	Reorderable list adaptor.
 <a href="#">ItemInsertedEventArgs</a>	Arguments which are passed to the <a href="#">ItemInsertedEvent</a> .

	<a href="#">ItemMovedEventArgs</a>	Arguments which are passed to the <a href="#">ItemMovedEvent</a> .
	<a href="#">ItemMovingEventArgs</a>	Arguments which are passed to the <a href="#">ItemMovingEvent</a> .
	<a href="#">ItemRemovingEventArgs</a>	Arguments which are passed to the <a href="#">ItemRemovingEvent</a> .
	<a href="#">ReorderableListControl</a>	Base class for custom controls.
	<a href="#">ReorderableListGUI</a>	Utility class for dragging lists.
	<a href="#">ReorderableListStyles</a>	Styles for the <a href="#">ReorderableList</a> .
	<a href="#">SerializedPropertyAdaptor</a>	Reorderable list adaptor for array property.

## Interfaces

Interface	Description
 <a href="#">IElementAdderTContext</a>	Interface for an object which adds elements of the type <i>TContext</i> .
 <a href="#">IElementAdderMenu</a>	Interface for a menu item.
 <a href="#">IElementAdderMenuBuilderTContext</a>	Interface for building menus.
 <a href="#">IElementAdderMenuCommandTContext</a>	Interface for a menu command.
 <a href="#">IReorderableListAdaptor</a>	Adaptor allowing to reorder data.

↳	<a href="#">IReorderableListDropTarget</a>	Can be implemented by drop insertion or
---	--	---

## ► Delegates

Delegate	Description
 <a href="#">AddMenuClickedEventHandler</a>	An event handler which is invoked when the "Add Menu" button is clicked.
 <a href="#">ItemInsertedEventHandler</a>	An event handler which is invoked after new list item is inserted.
 <a href="#">ItemMovedEventHandler</a>	An event handler which is invoked after a list item is moved.
 <a href="#">ItemMovingEventHandler</a>	An event handler which is invoked before a list item is

moved.

 	<a href="#">ItemRemovingEventHandler</a>	An event handler which is invoked before a list item is removed.
 	<a href="#">ReorderableListControlDrawEmpty</a>	Invoked to draw content for empty list.
 	<a href="#">ReorderableListControlDrawEmptyAbsolute</a>	Invoked to draw content for empty list with absolute positioning.
 	<a href="#">ReorderableListControlItemDrawerT</a>	Invoked to draw list item.

## ▪ **Enumerations**

Enumeration	Description
 	<a href="#">ReorderableListFlags</a> Additional flags which can be passed into reorderable list field.

# AddMenuClickedEventArgs Class

Arguments which are passed to [AddMenuClickedEventHandler](#).

## ► Inheritance Hierarchy

**SystemObject SystemEventArgs**

Rotorz.ReorderableListAddMenuClickedEventArgs

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public sealed class AddMenuClickedEventArgs : Eve
```

The [AddMenuClickedEventArgs](#) type exposes the following members.

## ► Constructors

	Name	Description
≡	<a href="#">AddMenuClickedEventArgs</a>	Initializes a new instance of <a href="#">ItemMovedEventArgs</a> .

[Top](#)

## ► Properties

	Name	Description
--	------	-------------



### Adaptor

Gets adaptor to reorderable list container.



### ButtonPosition

Gets position of the add menu button.

[Top](#)

## ► See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# AddMenuClickedEventArgs Constructor

Initializes a new instance of [ItemMovedEventArgs](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public AddMenuClickedEventArgs(  
    IReorderableListAdapter adaptor,  
    Rect buttonPosition  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)  
Reorderable list adaptor.

*buttonPosition*

Type: [Rect](#)  
Position of the add menu button.

## ► See Also

[Reference](#)

[AddMenuClickedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)



# AddMenuClickedEventArgs Properties

The [AddMenuClickedEventArgs](#) type exposes the following members.

## ► Properties

	Name	Description
	<a href="#">Adaptor</a>	Gets adaptor to reorderable list container.
	<a href="#">ButtonPosition</a>	Gets position of the add menu button.

[Top](#)

## ► See Also

### Reference

[AddMenuClickedEventArgs Class](#)  
[Rotorz.ReorderableList Namespace](#)

# AddMenuClickedEventArgsAdaptor Property

Gets adaptor to reorderable list container.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public IReorderableListAdapter Adaptor { get; }
```

Property Value

Type: [IReorderableListAdapter](#)

## ► See Also

Reference

[AddMenuClickedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# AddMenuClickedEventArgsButtonPosition Property

Gets position of the add menu button.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public Rect ButtonPosition { get; }
```

Property Value

Type: **Rect**

## ► See Also

Reference

[AddMenuClickedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# AddMenuClickedEventHandler Delegate

An event handler which is invoked when the "Add Menu" button is clicked.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public delegate void AddMenuClickedEventHandler(  
    Object sender,  
    AddMenuClickedEventArgs args  
)
```

## Parameters

*sender*

Type: **SystemObject**

Object which raised event.

*args*

Type: [Rotorz.ReorderableListAddMenuClickedEventArgs](#)

Event arguments.

## ► See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)



# ElementAdderMenuBuilder Class

Factory methods that create [IElementAdderMenuBuilderTContext](#) instances that can then be used to build element adder menus.

## ► Inheritance Hierarchy

[SystemObject](#) [Rotorz.ReorderableListElementAdderMenuBuilder](#)

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static class ElementAdderMenuBuilder
```

## ► Methods

Name	Description
   <a href="#">ForTContext</a>	Gets a <a href="#">IElementAdderMenuBuilderTContext</a> to build an element adder menu for a context object of the type <i>TContext</i> .
   <a href="#">ForTContext(Type)</a>	Gets a <a href="#">IElementAdderMenuBuilderTContext</a> to build an element adder menu for a context object of the type <i>TContext</i> .

[Top](#)

## Examples

The following example demonstrates how to build and display a menu which allows the user to add elements to a given context object upon clicking a button:

C#    UnityScript

[Copy](#)

```
public class ShoppingListElementAdder : IElementAdder
{
    public ShoppingListElementAdder(ShoppingList
        Object = shoppingList;
    }

    public ShoppingList Object { get; private set; }

    public bool CanAddElement(Type type) {
        return true;
    }

    public object AddElement(Type type) {
        var instance = Activator.CreateInstance(type);
        shoppingList.Add((ShoppingItem)instance);
        return instance;
    }
}

private void DrawAddMenuButton(ShoppingList shoppingList)
{
    var content = new GUIContent("Add Menu");
    Rect position = GUILayoutUtility.GetRect(content);
    if (GUI.Button(position, content)) {
        var builder = ElementAdderMenuBuilder.For(shoppingList);
        builder.SetElementAdder(new ShoppingListElementAdder());
        var menu = builder.GetMenu();
        menu.DropDown(buttonPosition);
    }
}
```

## ▲ See Also

Reference

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ElementAdderMenuBuilder Methods

## ◀ Methods

Name	Description
  <a href="#">ForTContext</a>	Gets a <a href="#">IElementAdderMenuBuilderTContext</a> to build an element adder menu for a context object of the type <i>TContext</i> .
  <a href="#">ForTContext(Type)</a>	Gets a <a href="#">IElementAdderMenuBuilderTContext</a> to build an element adder menu for a context object of the type <i>TContext</i> .

---

[Top](#)

## ◀ See Also

[Reference](#)

[ElementAdderMenuBuilder Class](#)

[Rotorz.ReorderableList Namespace](#)

# ElementAdderMenuBuilderForMethod

## ► Overload List

Name	Description
  <a href="#">ForTContext</a>	Gets a <a href="#">IElementAdderMenuBuilderTContext</a> to build an element adder menu for a context object of the type <i>TContext</i> .
  <a href="#">ForTContext(Type)</a>	Gets a <a href="#">IElementAdderMenuBuilderTContext</a> to build an element adder menu for a context object of the type <i>TContext</i> .

---

[Top](#)

## ► See Also

[Reference](#)

[ElementAdderMenuBuilder Class](#)

[Rotorz.ReorderableList Namespace](#)

# ElementAdderMenuBuilderForTContext Method

Gets a [IElementAdderMenuBuilderTContext](#) to build an element adder menu for a context object of the type *TContext*.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static IElementAdderMenuBuilder<TContext>
```

### Type Parameters

*TContext*

Type of the context object that elements can be added to.

### Return Value

Type: [IElementAdderMenuBuilderTContext](#)

A new [IElementAdderMenuBuilderTContext](#) instance.

## ► See Also

### Reference

[ElementAdderMenuBuilder Class](#)

For Overload

[Rotorz.ReorderableList Namespace](#)

[IElementAdderMenuBuilderTContextSetContractType\(Type\)](#)



# ElementAdderMenuBuilderForTContext<TContext>.GetElementAdderMenuBuilder Method (Type)

Gets a [IElementAdderMenuBuilder<TContext>](#) to build an element adder menu for a context object of the type *TContext*.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static IElementAdderMenuBuilder<TContext>
    Type contractType
)
```

## Parameters

*contractType*

Type: [SystemType](#)

Contract type of addable elements.

## Type Parameters

*TContext*

Type of the context object that elements can be added to.

## Return Value

Type: [IElementAdderMenuBuilder<TContext>](#)

A new [IElementAdderMenuBuilder<TContext>](#) instance.

## ► See Also

## Reference

[ElementAdderMenuBuilder Class](#)

For Overload

[Rotorz.ReorderableList Namespace](#)

[IElementAdderMenuBuilderTContextSetContractType\(Type\)](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ElementAdderMenuCommandAttribute Class

Annotate [IElementAdderMenuCommandTContext](#) implementations with a [ElementAdderMenuCommandAttribute](#) to associate it with the contract type of addable elements.

## ► Inheritance Hierarchy

**SystemObject SystemAttribute**

Rotorz.ReorderableListElementAdderMenuCommandAttribute

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public sealed class ElementAdderMenuCommandAttribute
```

The [ElementAdderMenuCommandAttribute](#) type exposes the following members.

## ► Constructors

Name	Description
 <a href="#">ElementAdderMenuCommandAttribute</a>	Initializes a new instance of the <a href="#">ElementAdderMenuCommandAttribute</a> class.

[Top](#)

## ◀ Methods

	Name	Description
	<b>Equals</b>	Returns a value that indicates whether this instance is equal to a specified object. (Inherited from <b>Attribute</b> .)
	<b>GetHashCode</b>	Returns the hash code for this instance. (Inherited from <b>Attribute</b> .)
	<b>IsDefaultAttribute</b>	When overridden in a derived class, indicates whether the value of this instance is the default value for the derived class. (Inherited from <b>Attribute</b> .)
	<b>Match</b>	When overridden in a derived class, returns a value that indicates whether this instance equals a specified object. (Inherited from <b>Attribute</b> .)

[Top](#)

## ◀ Properties

	Name	Description
	<b>ContractType</b>	Gets the contract type of addable elements.
	<b>TypeId</b>	When implemented in a derived class, gets a unique identifier for this <b>Attribute</b> .

(Inherited from **Attribute**.)

[Top](#)

## ▲ Examples

The following source code demonstrates how to add a helper menu command to the add element menu of a shopping list:

C#    UnityScript

[Copy](#)

```
[ElementAdderMenuCommand(typeof(ShoppingItem))]  
public class AddFavoriteShoppingItemsCommand : IE  
    public AddFavoriteShoppingItemsCommand() {  
        Content = new GUIContent("Add Favorite It  
    }  
  
    public GUIContent Content { get; private set;  
  
    public bool CanExecute(IElementAdder<Shopping  
        return true;  
    }  
    public void Execute(IElementAdder<ShoppingList  
        // TODO: Add favorite items to the shopp  
    }  
}
```

## ▲ See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

# ElementAdderMenuCommandAttribute Constructor

Initializes a new instance of the [ElementAdderMenuCommandAttribute](#) class.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public ElementAdderMenuCommandAttribute(  
    Type contractType  
)
```

### Parameters

*contractType*

Type: **SystemType**

Contract type of addable elements.

## ► See Also

Reference

[ElementAdderMenuCommandAttribute Class](#)

[Rotorz.ReorderableList Namespace](#)

# ElementAdderMenuCommandAttribute Methods

The [ElementAdderMenuCommandAttribute](#) type exposes the following members.

## ▪ Methods

	Name	Description
	<b>Equals</b>	Returns a value that indicates whether this instance is equal to a specified object. (Inherited from <a href="#">Attribute</a> .)
	<b>GetHashCode</b>	Returns the hash code for this instance. (Inherited from <a href="#">Attribute</a> .)
	<b>IsDefaultAttribute</b>	When overridden in a derived class, indicates whether the value of this instance is the default value for the derived class. (Inherited from <a href="#">Attribute</a> .)
	<b>Match</b>	When overridden in a derived class, returns a value that indicates whether this instance equals a specified object. (Inherited from <a href="#">Attribute</a> .)

[Top](#)

## ◀ See Also

### Reference

[ElementAdderMenuCommandAttribute Class](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ElementAdderMenuCommandAttribute Properties

The [ElementAdderMenuCommandAttribute](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">ContractType</a>	Gets the contract type of addable elements.
	<a href="#">TypeId</a>	When implemented in a derived class, gets a unique identifier for this <b>Attribute</b> . (Inherited from <b>Attribute</b> .)

[Top](#)

## See Also

### Reference

[ElementAdderMenuCommandAttribute Class](#)  
[Rotorz.ReorderableList Namespace](#)

# ElementAdderMenuCommandAttribute Property

Gets the contract type of addable elements.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public Type ContractType { get; }
```

Property Value

Type: **Type**

## ► See Also

Reference

[ElementAdderMenuCommandAttribute Class](#)

[Rotorz.ReorderableList Namespace](#)

# ElementAdderMeta Class

Provides meta information which is useful when creating new implementations of the [IElementAdderMenuBuilderTContext](#) interface.

## ► Inheritance Hierarchy

[SystemObject](#) [Rotorz.ReorderableListElementAdderMeta](#)

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static class ElementAdderMeta
```

The [ElementAdderMeta](#) type exposes the following members.

## ► Methods

Name	Description
  <a href="#">GetConcreteElementTypes(Type)</a>	Gets an array of all the element types that implement specified <i>contractType</i>
  <a href="#">GetConcreteElementTypes(Type, FuncType, Boolean)</a>	Gets a filtered array of element types that implement specified <i>contractType</i>
  <a href="#">GetMenuCommandsTContext</a>	Gets an array of <a href="#">IElementAdderMenuBuilderTContext</a> instances that are associated with this class

specified *contractType*

---



[GetMenuCommandTypesTContext](#)

Gets an array of the [IElementAdderMenuItem](#) types that are associated with the specified *contractType*

---

[Top](#)

## See Also

Reference

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ElementAdderMeta Methods

The [ElementAdderMeta](#) type exposes the following members.

## Methods

Name	Description
<a href="#">GetConcreteElementTypes(Type)</a>	Gets an array of all the element types that implement specified <i>contractType</i>
<a href="#">GetConcreteElementTypes(Type, FuncType, Boolean)</a>	Gets a filtered array of element types that implement specified <i>contractType</i>
<a href="#">GetMenuCommandsTContext</a>	Gets an array of <a href="#">IElementAdderMenuCommand</a> instances that are associated with specified <i>contractType</i>
<a href="#">GetMenuCommandTypesTContext</a>	Gets an array of the <a href="#">IElementAdderMenuCommand</a> types that are associated with specified <i>contractType</i>

[Top](#)

## See Also

### Reference

[ElementAdderMeta Class](#)

[Rotorz.ReorderableList Namespace](#)



# ElementAdderMetaGetConcreteEler Method

## ► Overload List

	Name	Description
 	<a href="#">GetConcreteElementTypes(Type)</a>	Gets an array of all the concrete element types that implement the specified <i>contractType</i> .
 	<a href="#">GetConcreteElementTypes(Type, FuncType, Boolean)</a>	Gets a filtered array of the concrete element types that implement the specified <i>contractType</i> .

[Top](#)

## ► See Also

Reference

[ElementAdderMeta Class](#)

[Rotorz.ReorderableList Namespace](#)

# ElementAdderMetaGetConcreteEler Method (Type)

Gets an array of all the concrete element types that implement the specified *contractType*.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static Type[] GetConcreteElementTypes(  
    Type contractType  
)
```

### Parameters

*contractType*

Type: **SystemType**

Contract type of addable elements.

### Return Value

Type: **Type**

An array of zero or more concrete element types.

## ► Exceptions

Exception	Condition
<b>ArgumentNullException</b>	If <i>contractType</i> is <b>null</b> .

## ▲ See Also

### Reference

[ElementAdderMeta Class](#)

[GetConcreteElementTypes Overload](#)

[Rotorz.ReorderableList Namespace](#)

[ElementAdderMeta.GetConcreteElementTypes\(Type, FuncType, Boolean\)](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ElementAdderMetaGetConcreteElementTypes Method (Type, Func<Type, Boolean>, Func<Type, bool>[])

Gets a filtered array of the concrete element types that implement the specified *contractType*.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static Type[] GetConcreteElementTypes(  
    Type contractType,  
    Func<Type, bool>[] filters  
)
```

## Parameters

*contractType*

Type: **SystemType**

Contract type of addable elements.

*filters*

Type: **SystemFuncType, Boolean**

An array of zero or more filters.

## Return Value

Type: **Type**

An array of zero or more concrete element types.

## ► Exceptions

Exception

Condition

---

**ArgumentNullException**

If *contractType* is **null**.

---

## Remarks

A type is excluded from the resulting array when one or more of the specified *filters* returns a value of **false**.

## See Also

### Reference

[ElementAdderMeta Class](#)

[GetConcreteElementTypes Overload](#)

[Rotorz.ReorderableList Namespace](#)

[ElementAdderMetaGetConcreteElementTypes\(Type\)](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ElementAdderMetaGetMenuCommand Method

Gets an array of [IElementAdderMenuCommand<TContext>](#) instances that are associated with the specified *contractType*.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static IElementAdderMenuCommand<TContext>[]  
    Type contractType  
)
```

## Parameters

*contractType*

Type: [SystemType](#)

Contract type of addable elements.

## Type Parameters

*TContext*

Type of the context object that elements can be added to.

## Return Value

Type: [IElementAdderMenuCommand<TContext>](#)

An array containing zero or more

[IElementAdderMenuCommand<TContext>](#) instances.

## ▪ Exceptions

Exception	Condition
<a href="#">ArgumentNullException</a>	If <i>contractType</i> is <code>null</code> .

## ▪ See Also

### Reference

[ElementAdderMeta Class](#)

[Rotorz.ReorderableList Namespace](#)

[ElementAdderMetaGetMenuCommandTypesTContext\(Type\)](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ElementAdderMetaGetMenuCommandTypes Method

Gets an array of the [IElementAdderMenuCommandTContext](#) types that are associated with the specified *contractType*.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static Type[] GetMenuCommandTypes<TContext>()
{
    Type contractType
}
```

## Parameters

*contractType*

Type: [SystemType](#)

Contract type of addable elements.

## Type Parameters

*TContext*

Type of the context object that elements can be added to.

## Return Value

Type: [Type](#)

An array containing zero or more [Type](#).

## ► Exceptions

Exception	Condition
<b>ArgumentNullException</b>	If <i>contractType</i> is <b>null</b> .

## See Also

### Reference

[ElementAdderMeta Class](#)

[Rotorz.ReorderableList Namespace](#)

[ElementAdderMetaGetMenuCommandsTContext\(Type\)](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# GenericListAdapter<T> Class

Reorderable list adaptor for generic list.

## ► Inheritance Hierarchy

**SystemObject** Rotorz.ReorderableListGenericListAdapter<T>

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public class GenericListAdapter<T> : IReorderable
```

### Type Parameters

T

Type of list element.

The **GenericListAdapter<T>** type exposes the following members.

## ► Constructors

	Name	Description
≡	<a href="#">GenericListAdapter&lt;T&gt;</a>	Initializes a new instance of <b>GenericListAdapter&lt;T&gt;</b> .

[Top](#)

## ▪ Methods

	Name	Description
≡■	Add	Add new element at end of list.
≡■	BeginGUI	Occurs before any list items are drawn.
≡■	CanDrag	Determines whether an item can be reordered by dragging mouse.
≡■	CanRemove	Determines whether an item can be removed from list.
≡■	Clear	Clear all elements from list.
≡■	DrawItem	Draws main interface for a list item.
≡■	DrawItemBackground	Draws background of a list item.
≡■	Duplicate	Duplicate existing element.
≡■	EndGUI	Occurs after all list items have been drawn.
≡■	GetItemHeight	Gets height of list item in pixels.
≡■	Insert	Insert new element at specified index.
≡■	Move	Move element from source index to destination index.



## Remove

Remove element at specified index.

[Top](#)

## Fields

	Name	Description
◆	<a href="#">FixedItemHeight</a>	Fixed height of each list item.

[Top](#)

## Properties

	Name	Description
◆	<a href="#">Count</a>	Gets count of elements in list.
◆	<a href="#">Item</a>	Gets element from list.
◆	<a href="#">List</a>	Gets the underlying list data structure.

[Top](#)

## Remarks

This adaptor can be subclassed to add special logic to item height calculation. You may want to implement a custom adaptor class where specialised functionality is needed.

List elements which implement the **ICloneable** interface are cloned using that interface upon duplication; otherwise the item value or reference is simply copied.

## See Also

### Reference

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# GenericListAdapterT Constructor

Initializes a new instance of [GenericListAdapterT](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public GenericListAdapter(
    IList<T> list,
    ReorderableListControlItemDrawer<T> itemDrawer,
    float itemHeight
)
```

## Parameters

*list*

Type: [System.Collections.GenericIListT](#)

The list which can be reordered.

*itemDrawer*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawerT](#)

Callback to draw list item.

*itemHeight*

Type: [SystemSingle](#)

Height of list item in pixels.

## ► See Also

[Reference](#)

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# GenericListAdapterT Fields

The [GenericListAdapterT](#) generic type exposes the following members.

## Fields

	Name	Description
◆	<a href="#">FixedItemHeight</a>	Fixed height of each list item.

[Top](#)

## See Also

[Reference](#)

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# GenericListAdapterT<sub>TFixedItemHeight</sub> Field

Fixed height of each list item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public float FixedItemHeight
```

Field Value

Type: **Single**

## ► See Also

Reference

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

# GenericListAdapterT Methods

The [GenericListAdapterT](#) generic type exposes the following members.

## Methods

	Name	Description
≡	<a href="#">Add</a>	Add new element at end of list.
≡	<a href="#">BeginGUI</a>	Occurs before any list items are drawn.
≡	<a href="#">CanDrag</a>	Determines whether an item can be reordered by dragging mouse.
≡	<a href="#">CanRemove</a>	Determines whether an item can be removed from list.
≡	<a href="#">Clear</a>	Clear all elements from list.
≡	<a href="#">DrawItem</a>	Draws main interface for a list item.
≡	<a href="#">DrawItemBackground</a>	Draws background of a list item.
≡	<a href="#">Duplicate</a>	Duplicate existing element.
≡	<a href="#">EndGUI</a>	Occurs after all list items have been drawn.
≡	<a href="#">GetItemHeight</a>	Gets height of list item in pixels.

---

	<a href="#">Insert</a>	Insert new element at specified index.
	<a href="#">Move</a>	Move element from source index to destination index.
	<a href="#">Remove</a>	Remove element at specified index.

---

[Top](#)

## ▲ See Also

### Reference

[GenericListAdaptorT Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# GenericListAdapterTAdd Method

Add new element at end of list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    [UnityScript](#)

[Copy](#)

```
public virtual void Add()
```

Implements

[IReorderableListAdapterAdd](#)

## ► See Also

[Reference](#)

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

# GenericListAdapterTBeginGUI Method

Occurs before any list items are drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void BeginGUI()
```

Implements

[IReorderableListAdaptorBeginGUI](#)

## ► Remarks

This method is only used to handle GUI repaint events.

## ► See Also

Reference

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

# GenericListAdapter{T}CanDrag Method

Determines whether an item can be reordered by dragging mouse.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#    UnityScript

Copy

```
public virtual bool CanDrag(  
    int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index for list element.

### Return Value

Type: **Boolean**

A value of **true** if item can be dragged; otherwise **false**.

### Implements

[IReorderableListAdapterCanDrag\(Int32\)](#)

## ◀ Remarks

This should be a light-weight method since it will be used to determine whether grab handle should be included for each item in a reorderable list.

Please note that returning a value of `false` does not prevent movement on list item since other draggable items can be moved around it.

## ▲ See Also

Reference

[GenericListAdaptorT Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# GenericListAdapter{T}CanRemove Method

Determines whether an item can be removed from list.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public virtual bool CanRemove(  
    int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index for list element.

### Return Value

Type: **Boolean**

A value of **true** if item can be removed; otherwise **false**.

### Implements

[IReorderableListAdapterCanRemove\(Int32\)](#)

## ► Remarks

This should be a light-weight method since it will be used to determine whether remove button should be included for each item in list.

This is redundant when [HideRemoveButtons](#) is specified.

## ◀ See Also

[Reference](#)

[GenericListAdaptorT Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# GenericListAdapterTClear Method

Clear all elements from list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void Clear()
```

Implements

[IReorderableListAdapterClear](#)

## ► See Also

[Reference](#)

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

# GenericListAdapter<TDrawItem> Method

Draws main interface for a list item.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public virtual void DrawItem(  
    Rect position,  
    int index  
)
```

### Parameters

*position*

Type: **Rect**

Position in GUI.

*index*

Type: **SystemInt32**

Zero-based index of array element.

### Implements

[IReorderableListAdaptorDrawItem\(Rect, Int32\)](#)

## ► Remarks

This method is used to handle all GUI events.

## ◀ See Also

### Reference

[GenericListAdaptorT Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# GenericListAdapter<T>.DrawItemBackground Method

Draws background of a list item.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void DrawItemBackground(  
    Rect position,  
    int index  
)
```

### Parameters

*position*

Type: **Rect**

Total position of list element in GUI.

*index*

Type: **SystemInt32**

Zero-based index of array element.

### Implements

[IReorderableListAdapter<T>.DrawItemBackground\(Rect, Int32\)](#)

## ► Remarks

This method is only used to handle GUI repaint events.

Background of list item spans a slightly larger area than the main interface that is drawn by [DrawItem\(Rect, Int32\)](#) since it is drawn

behind the grab handle.

## ◀ See Also

[Reference](#)

[GenericListAdaptorT Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# GenericListAdapter TDuplicate Method

Duplicate existing element.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void Duplicate(  
    int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index of list element.

### Implements

[IReorderableListAdapterDuplicate\(Int32\)](#)

## ► Remarks

Consider using the **ICloneable** interface to duplicate list elements where appropriate.

## ► See Also

[Reference](#)

GenericListAdaptorT Class  
Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# GenericListAdapterT::EndGUI Method

Occurs after all list items have been drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void EndGUI()
```

Implements

[IReorderableListAdaptor::EndGUI](#)

## ► Remarks

This method is only used to handle GUI repaint events.

## ► See Also

Reference

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

# GenericListAdapterT`GetItemHeight` Method

Gets height of list item in pixels.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#    UnityScript

[Copy](#)

```
public virtual float GetItemHeight(  
    int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index of array element.

### Return Value

Type: **Single**

Measurement in pixels.

### Implements

[IReorderableListAdapter.GetItemHeight\(Int32\)](#)

## ◀ See Also

[Reference](#)

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# GenericListAdapterTInsert Method

Insert new element at specified index.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void Insert(  
    int index  
)
```

## Parameters

*index*

Type: **SystemInt32**

Zero-based index for list element.

## Implements

[IReorderableListAdapterInsert\(Int32\)](#)

## ► See Also

### Reference

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

# GenericListAdapterTMove Method

Move element from source index to destination index.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void Move(  
    int sourceIndex,  
    int destIndex  
)
```

### Parameters

*sourceIndex*

Type: **SystemInt32**

Zero-based index of source element.

*destIndex*

Type: **SystemInt32**

Zero-based index of destination element.

### Implements

[IReorderableListAdaptorMove\(Int32, Int32\)](#)

## ► See Also

### Reference

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)



# GenericListAdapterT Remove Method

Remove element at specified index.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void Remove(  
    int index  
)
```

### Parameters

*index*

Type: [SystemInt32](#)

Zero-based index of list element.

### Implements

[IReorderableListAdaptorRemove\(Int32\)](#)

## ► See Also

[Reference](#)

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

# GenericListAdapterT Properties

The [GenericListAdapterT](#) generic type exposes the following members.

## Properties

	Name	Description
	<a href="#">Count</a>	Gets count of elements in list.
	<a href="#">Item</a>	Gets element from list.
	<a href="#">List</a>	Gets the underlying list data structure.

[Top](#)

## See Also

### Reference

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

# GenericListAdapterTCount Property

Gets count of elements in list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public int Count { get; }
```

Property Value

Type: [Int32](#)

Implements

[IReorderableListAdaptorCount](#)

## ► See Also

Reference

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

# GenericListAdapterTItem Property

Gets element from list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public T this[  
    int index  
] { get; }
```

## Parameters

*index*

Type: **SystemInt32**

Zero-based index of element.

## Return Value

Type: **T**

The element.

## ► See Also

### Reference

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

# GenericListAdapterTList Property

Gets the underlying list data structure.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    [UnityScript](#)

[Copy](#)

```
public IList<T> List { get; }
```

Property Value

Type: [IList](#)*T*

## ► See Also

[Reference](#)

[GenericListAdapterT Class](#)

[Rotorz.ReorderableList Namespace](#)

# IElementAdder<TContext> Interface

Interface for an object which adds elements to a context object of the type *TContext*.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public interface IElementAdder<TContext>
```

### Type Parameters

*TContext*

Type of the context object that elements can be added to.

The **IElementAdder<TContext>** type exposes the following members.

## ► Methods

	Name	Description
≡	<a href="#">AddElement</a>	Adds an element of the specified <i>type</i> to the associated context object.
≡	<a href="#">CanAddElement</a>	Determines whether a new element of the specified <i>type</i> can be added to the associated context object.

[Top](#)

## ► Properties

	Name	Description
	<a href="#">Object</a>	Gets the context object.

[Top](#)

## ► See Also

### Reference

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderTContext Methods

The [IElementAdderTContext](#) generic type exposes the following members.

## ▪ Methods

	Name	Description
	<a href="#">AddElement</a>	Adds an element of the specified <i>type</i> to the associated context object.
	<a href="#">CanAddElement</a>	Determines whether a new element of the specified <i>type</i> can be added to the associated context object.

[Top](#)

## ▪ See Also

### Reference

[IElementAdderTContext Interface](#)  
[Rotorz.ReorderableList Namespace](#)

# IElementAdderTContextAddElement Method

Adds an element of the specified *type* to the associated context object.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
Object AddElement(  
    Type type  
)
```

### Parameters

*type*

Type: **SystemType**

Type of element to add.

### Return Value

Type: **Object**

The new element.

## ► See Also

[Reference](#)

[IElementAdderTContext Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IElementAdderTContextCanAddElement Method

Determines whether a new element of the specified *type* can be added to the associated context object.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
bool CanAddElement(  
    Type type  
)
```

### Parameters

*type*

Type: **SystemType**

Type of element to add.

### Return Value

Type: **Boolean**

A value of [true](#) if an element of the specified type can be added; otherwise, a value of [false](#).

## ► See Also

Reference

[IElementAdderTContext Interface](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderTContext Properties

The [IElementAdderTContext](#) generic type exposes the following members.

## ► Properties

	Name	Description
	<a href="#">Object</a>	Gets the context object.

[Top](#)

## ► See Also

### Reference

[IElementAdderTContext Interface](#)  
[Rotorz.ReorderableList Namespace](#)

# IElementAdder{TContext}Object Property

Gets the context object.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
TContext Object { get; }
```

Property Value

Type: [TContext](#)

## ► See Also

Reference

[IElementAdder{TContext} Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IElementAdderMenu Interface

Interface for a menu interface.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public interface IElementAdderMenu
```

The [IElementAdderMenu](#) type exposes the following members.

## ► Methods

	Name	Description
	<a href="#">DropDown</a>	Displays the drop-down menu inside an editor GUI.

[Top](#)

## ► Properties

	Name	Description
	<a href="#">IsEmpty</a>	Gets a value indicating whether the menu contains any items.

[Top](#)

## ► See Also

## Reference

### [Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderMenu Methods

The [IElementAdderMenu](#) type exposes the following members.

## ▲ Methods

	Name	Description
	<a href="#">DropDown</a>	Displays the drop-down menu inside an editor GUI.

[Top](#)

## ▲ See Also

[Reference](#)

[IElementAdderMenu Interface](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderMenuDropDown Method

Displays the drop-down menu inside an editor GUI.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void DropDown(  
    Rect position  
)
```

### Parameters

*position*

Type: **Rect**

Position of menu button in the GUI.

## ► Remarks

This method should only be used during **OnGUI** and **OnSceneGUI** events; for instance, inside an editor window, a custom inspector or scene view.

## ► See Also

Reference

[IElementAdderMenu Interface](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderMenu Properties

The [IElementAdderMenu](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">IsEmpty</a>	Gets a value indicating whether the menu contains any items.

[Top](#)

## See Also

Reference

[IElementAdderMenu Interface](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderMenusEmpty Property

Gets a value indicating whether the menu contains any items.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
bool IsEmpty { get; }
```

Property Value

Type: **Boolean**

[true](#) if the menu contains one or more items; otherwise, [false](#).

## ► See Also

[Reference](#)

[IElementAdderMenu Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IElementAdderMenuBuilder<TContext> Interface

Interface for building an [IElementAdderMenu](#).

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public interface IElementAdderMenuBuilder<TConte>
```

### Type Parameters

#### *TContext*

Type of the context object that elements can be added to.

The [IElementAdderMenuBuilder<TContext>](#) type exposes the following members.

## ► Methods

Name	Description
<a href="#">AddCustomCommand</a>	Adds a custom command to the menu.
<a href="#">AddTypeFilter</a>	Adds a filter function which determines whether types can be included or whether

		they need to be excluded.
≡	<a href="#">GetMenu</a>	Builds and returns a new <a href="#">IElementAdderMenu</a> instance.
≡	<a href="#">SetContractType</a>	Sets contract type of the elements that can be included in the <a href="#">IElementAdderMenu</a> . Only non-abstract class types that are assignable from the <i>contractType</i> will be included in the built menu.
≡	<a href="#">SetElementAdder</a>	Set the <a href="#">IElementAdderTContext</a> implementation which is used when adding new elements to the context object.
≡	<a href="#">SetTypeDisplayNameFormatter</a>	Set the function that formats the display of type names in the user interface.

[Top](#)

## ▲ See Also

### Reference

[Rotorz.ReorderableList Namespace](#)



# IElementAdderMenuBuilderTContext Methods

The [IElementAdderMenuBuilderTContext](#) generic type exposes the following members.

## ▪ Methods

Name	Description
<a href="#">AddCustomCommand</a>	Adds a custom command to the menu.
<a href="#">AddTypeFilter</a>	Adds a filter function which determines whether types can be included or whether they need to be excluded.
<a href="#">GetMenu</a>	Builds and returns a new <a href="#">IElementAdderMenu</a> instance.
<a href="#">SetContractType</a>	Sets contract type of the elements that can be included in the <a href="#">IElementAdderMenu</a> . Only non-abstract class types that are assignable from the <i>contractType</i> will be included in the built menu.

---

 <a href="#">SetElementAdder</a>	Set the <a href="#">IElementAdderTContext</a> implementation which is used when adding new elements to the context object.
 <a href="#">SetTypeDisplayNameFormatter</a>	Set the function that formats the display of type names in the user interface.

---

[Top](#)

## ▲ See Also

### Reference

[IElementAdderMenuBuilderTContext Interface](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderMenuBuilderTContext Method

Adds a custom command to the menu.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void AddCustomCommand(  
    IElementAdderMenuCommand<TContext> command  
)
```

### Parameters

*command*

Type: [Rotorz.ReorderableListIElementAdderMenuCommandTContext](#)  
The custom command.

## ► Exceptions

Exception	Condition
<a href="#">ArgumentNullException</a>	If <i>command</i> is <a href="#">null</a> .

## ► See Also

Reference

[IElementAdderMenuBuilderTContext Interface](#)

## Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderMenuBuilder<TContext>.Method

Adds a filter function which determines whether types can be included or whether they need to be excluded.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void AddTypeFilter(  
    Func<Type, bool> typeFilter  
)
```

### Parameters

*typeFilter*

Type: **SystemFuncType, Boolean**

Filter function.

## ► Exceptions

Exception	Condition
<b>ArgumentNullException</b>	If <i>typeFilter</i> is <b>null</b> .

## ► Remarks

If a filter function returns a value of **false** then that type will not be included in the menu interface.

## ◀ See Also

### Reference

[IElementAdderMenuBuilderTContext Interface](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderMenuBuilderTContext Method

Builds and returns a new [IElementAdderMenu](#) instance.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
IElementAdderMenu GetMenu()
```

### Return Value

Type: [IElementAdderMenu](#)

A new [IElementAdderMenu](#) instance each time the method is invoked.

## ► See Also

### Reference

[IElementAdderMenuBuilderTContext Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IElementAdderMenuBuilderTContext Method

Sets contract type of the elements that can be included in the [IElementAdderMenu](#). Only non-abstract class types that are assignable from the *contractType* will be included in the built menu.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void SetContractType(  
    Type contractType  
)
```

### Parameters

*contractType*

Type: **SystemType**

Contract type of addable elements.

## ► See Also

### Reference

[IElementAdderMenuBuilderTContext Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IElementAdderMenuBuilderTContext Method

Set the [IElementAdderTContext](#) implementation which is used when adding new elements to the context object.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
void SetElementAdder(  
    IElementAdder<TContext> elementAdder  
)
```

### Parameters

*elementAdder*

Type: [Rotorz.ReorderableListIElementAdderTContext](#)  
Element adder.

## ► Remarks

Specify a value of `null` for *elementAdder* to prevent the selection of any types.

## ► See Also

Reference

[IElementAdderMenuBuilderTContext Interface](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderMenuBuilderTContext Method

Set the function that formats the display of type names in the user interface.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
void SetTypeDisplayNameFormatter(  
    Func<Type, string> formatter  
)
```

### Parameters

*formatter*

Type: **SystemFuncType, String**

Function that formats display name of type; or **null**.

## ► Remarks

Specify a value of **null** for *formatter* to assume the default formatting function.

## ► See Also

Reference

[IElementAdderMenuBuilderTContext Interface](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderMenuCommandTContext Interface

Interface for a menu command that can be included in an [IElementAdderMenu](#) either by annotating an implementation of the [IElementAdderMenuCommandTContext](#) interface with [ElementAdderMenuCommandAttribute](#) or directly by calling [AddCustomCommand\(IElementAdderMenuCommandTContext\)](#).

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public interface IElementAdderMenuCommand<TConte>
```

### Type Parameters

#### *TContext*

Type of the context object that elements can be added to.

The [IElementAdderMenuCommandTContext](#) type exposes the following members.

## ► Methods

	Name	Description
	<a href="#">CanExecute</a>	Determines whether the command can be executed.

---

Execute	Executes the command.
---------	-----------------------

---

[Top](#)

## ► Properties

	Name	Description
	<a href="#">Content</a>	Gets the content of the menu command.

---

[Top](#)

## ► See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IElementAdderMenuCommandTContext Methods

The [IElementAdderMenuCommandTContext](#) generic type exposes the following members.

## ▪ Methods

	Name	Description
≡	<a href="#">CanExecute</a>	Determines whether the command can be executed.
≡	<a href="#">Execute</a>	Executes the command.

[Top](#)

## ▪ See Also

### Reference

[IElementAdderMenuCommandTContext Interface](#)  
[Rotorz.ReorderableList Namespace](#)

# IElementAdderMenuCommandTContext Method

Determines whether the command can be executed.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
bool CanExecute(  
    IElementAdder<TContext> elementAdder  
)
```

### Parameters

*elementAdder*

Type: [Rotorz.ReorderableListIElementAdderTContext](#)

The associated element adder provides access to the *TContext* instance.

### Return Value

Type: **Boolean**

A value of `true` if the command can execute; otherwise, `false`.

## ► See Also

### Reference

[IElementAdderMenuCommandTContext Interface](#)

[Rotorz.ReorderableList Namespace](#)



# IElementAdderMenuCommandTContext Method

Executes the command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void Execute(  
    IElementAdder<TContext> elementAdder  
)
```

### Parameters

*elementAdder*

Type: [Rotorz.ReorderableListIElementAdderTContext](#)

The associated element adder provides access to the *TContext* instance.

## ► See Also

[Reference](#)

[IElementAdderMenuCommandTContext Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IElementAdderMenuCommandTContext Properties

The [IElementAdderMenuCommandTContext](#) generic type exposes the following members.

## Properties

	Name	Description
	<a href="#">Content</a>	Gets the content of the menu command.

[Top](#)

## See Also

[Reference](#)

[IElementAdderMenuCommandTContext Interface](#)  
[Rotorz.ReorderableList Namespace](#)

# IElementAdderMenuCommandTContext Property

Gets the content of the menu command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
GUIContent Content { get; }
```

Property Value

Type: **GUIContent**

## ► See Also

Reference

[IElementAdderMenuCommandTContext Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IReorderableListAdapter Interface

Adaptor allowing reorderable list control to interface with list data.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public interface IReorderableListAdapter
```

The [IReorderableListAdapter](#) type exposes the following members.

## ► Methods

	Name	Description
≡	<a href="#">Add</a>	Add new element at end of list.
≡	<a href="#">BeginGUI</a>	Occurs before any list items are drawn.
≡	<a href="#">CanDrag</a>	Determines whether an item can be reordered by dragging mouse.
≡	<a href="#">CanRemove</a>	Determines whether an item can be removed from list.
≡	<a href="#">Clear</a>	Clear all elements from list.
≡	<a href="#">DrawItem</a>	Draws main interface for a list

item.

---

≡	<a href="#">DrawItemBackground</a>	Draws background of a list item.
≡	<a href="#">Duplicate</a>	Duplicate existing element.
≡	<a href="#">EndGUI</a>	Occurs after all list items have been drawn.
≡	<a href="#">GetItemHeight</a>	Gets height of list item in pixels.
≡	<a href="#">Insert</a>	Insert new element at specified index.
≡	<a href="#">Move</a>	Move element from source index to destination index.
≡	<a href="#">Remove</a>	Remove element at specified index.

---

[Top](#)

## Properties

	Name	Description
	<a href="#">Count</a>	Gets count of elements in list.

---

[Top](#)

## See Also

Reference

[Rotorz.ReorderableList Namespace](#)

# IReorderableListAdaptor Methods

The [IReorderableListAdaptor](#) type exposes the following members.

## ▲ Methods

	Name	Description
≡	<a href="#">Add</a>	Add new element at end of list.
≡	<a href="#">BeginGUI</a>	Occurs before any list items are drawn.
≡	<a href="#">CanDrag</a>	Determines whether an item can be reordered by dragging mouse.
≡	<a href="#">CanRemove</a>	Determines whether an item can be removed from list.
≡	<a href="#">Clear</a>	Clear all elements from list.
≡	<a href="#">DrawItem</a>	Draws main interface for a list item.
≡	<a href="#">DrawItemBackground</a>	Draws background of a list item.
≡	<a href="#">Duplicate</a>	Duplicate existing element.
≡	<a href="#">EndGUI</a>	Occurs after all list items have been drawn.
≡	<a href="#">GetItemHeight</a>	Gets height of list item in pixels.

---

	<a href="#">Insert</a>	Insert new element at specified index.
	<a href="#">Move</a>	Move element from source index to destination index.
	<a href="#">Remove</a>	Remove element at specified index.

---

[Top](#)

## ▲ See Also

### Reference

[IReorderableListAdaptor Interface](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IReorderableListAdaptorAdd Method

Add new element at end of list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void Add()
```

## ► See Also

Reference

[IReorderableListAdaptor Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IReorderableListAdaptorBeginGUI Method

Occurs before any list items are drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void BeginGUI()
```

## ► Remarks

This method is only used to handle GUI repaint events.

## ► See Also

Reference

[IReorderableListAdaptor Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IReorderableListAdaptorCanDrag Method

Determines whether an item can be reordered by dragging mouse.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
bool CanDrag(  
            int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index for list element.

### Return Value

Type: **Boolean**

A value of **true** if item can be dragged; otherwise **false**.

## ► Remarks

This should be a light-weight method since it will be used to determine whether grab handle should be included for each item in a reorderable list.

Please note that returning a value of **false** does not prevent movement on list item since other draggable items can be moved around it.

## ◀ See Also

### Reference

[IReorderableListAdaptor Interface](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IReorderableListAdaptorCanRemove Method

Determines whether an item can be removed from list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
bool CanRemove(  
                int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index for list element.

### Return Value

Type: **Boolean**

A value of [true](#) if item can be removed; otherwise [false](#).

## ► Remarks

This should be a light-weight method since it will be used to determine whether remove button should be included for each item in list.

This is redundant when [HideRemoveButtons](#) is specified.

## ◀ See Also

### Reference

[IReorderableListAdaptor Interface](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IReorderableListAdaptorClear Method

Clear all elements from list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void Clear()
```

## ► See Also

[Reference](#)

[IReorderableListAdaptor Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IR reorderableListAdaptorDrawItem

## Method

Draws main interface for a list item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

### ► Syntax

C#    UnityScript

[Copy](#)

```
void DrawItem(  
    Rect position,  
    int index  
)
```

### Parameters

*position*

Type: **Rect**

Position in GUI.

*index*

Type: **SystemInt32**

Zero-based index of array element.

### ► Remarks

This method is used to handle all GUI events.

### ► See Also

Reference

**IReorderableListAdaptor Interface**  
**Rotorz.ReorderableList Namespace**

---

©2013-2015 Rotorz Limited. All rights reserved.

# IReorderableListAdaptorDrawItemBackground Method

Draws background of a list item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void DrawItemBackground(  
    Rect position,  
    int index  
)
```

### Parameters

*position*

Type: **Rect**

Total position of list element in GUI.

*index*

Type: **SystemInt32**

Zero-based index of array element.

## ► Remarks

This method is only used to handle GUI repaint events.

Background of list item spans a slightly larger area than the main interface that is drawn by [DrawItem\(Rect, Int32\)](#) since it is drawn behind the grab handle.

## ◀ See Also

### Reference

[IReorderableListAdaptor Interface](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IReorderableListAdaptorDuplicate Method

Duplicate existing element.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void Duplicate(  
    int index  
)
```

### Parameters

*index*

Type: [SystemInt32](#)

Zero-based index of list element.

## ► Remarks

Consider using the **ICloneable** interface to duplicate list elements where appropriate.

## ► See Also

Reference

[IReorderableListAdaptor Interface](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IReorderableListAdaptorEndGUI Method

Occurs after all list items have been drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void EndGUI()
```

## ► Remarks

This method is only used to handle GUI repaint events.

## ► See Also

Reference

[IReorderableListAdaptor Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IReorderableListAdaptor.GetItemHeight Method

Gets height of list item in pixels.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
float GetItemHeight(  
    int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index of array element.

### Return Value

Type: **Single**

Measurement in pixels.

## ► See Also

[Reference](#)

[IReorderableListAdaptor Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IReorderableListAdaptorInsert Method

Insert new element at specified index.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void Insert(  
            int index  
)
```

### Parameters

*index*

Type: [SystemInt32](#)

Zero-based index for list element.

## ► See Also

Reference

[IReorderableListAdaptor Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IReorderableListAdaptorMove Method

Move element from source index to destination index.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void Move(  
            int sourceIndex,  
            int destIndex  
)
```

### Parameters

*sourceIndex*

Type: **SystemInt32**

Zero-based index of source element.

*destIndex*

Type: **SystemInt32**

Zero-based index of destination element.

## ► See Also

[Reference](#)

[IReorderableListAdaptor Interface](#)

[Rotorz.ReorderableList Namespace](#)



# IReorderableListAdaptorRemove Method

Remove element at specified index.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void Remove(  
            int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index of list element.

## ► See Also

Reference

[IReorderableListAdaptor Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IReorderableListAdaptor Properties

The [IReorderableListAdaptor](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Count</a>	Gets count of elements in list.

[Top](#)

## See Also

Reference

[IReorderableListAdaptor Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IReorderableListAdaptorCount Property

Gets count of elements in list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
int Count { get; }
```

Property Value

Type: [Int32](#)

## ► See Also

Reference

[IReorderableListAdaptor Interface](#)

[Rotorz.ReorderableList Namespace](#)

# IReorderableListDropTarget Interface

Can be implemented along with [IReorderableListAdapter](#) when drop insertion or ordering is desired.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public interface IReorderableListDropTarget
```

The [IReorderableListDropTarget](#) type exposes the following members.

## ► Methods

	Name	Description
≡	<a href="#">CanDropInsert</a>	Determines whether an item is being dragged and that it can be inserted or moved by dropping somewhere into the reorderable list control.
≡	<a href="#">ProcessDropInsertion</a>	Processes the current drop insertion operation when <a href="#">CanDropInsert(Int32)</a> returns a value of <a href="#">true</a> to process, accept or cancel.

[Top](#)

## ► Remarks

This type of "drop" functionality can occur when the "drag" phase of the drag and drop operation was initiated elsewhere. For example, a custom [IReorderableListAdapter](#) could insert entirely new items by dragging and dropping from the Unity "Project" window.

## ► See Also

Reference

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IReorderableListDropTarget Methods

The [IReorderableListDropTarget](#) type exposes the following members.

## ▪ Methods

Name	Description
 <a href="#">CanDropInsert</a>	Determines whether an item is being dragged and that it can be inserted or moved by dropping somewhere into the reorderable list control.
 <a href="#">ProcessDropInsertion</a>	Processes the current drop insertion operation when <a href="#">CanDropInsert(Int32)</a> returns a value of <code>true</code> to process, accept or cancel.

[Top](#)

## ▪ See Also

### Reference

[IReorderableListDropTarget Interface](#)  
[Rotorz.ReorderableList Namespace](#)

# IReorderableListDropTargetCanDrop Method

Determines whether an item is being dragged and that it can be inserted or moved by dropping somewhere into the reorderable list control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
bool CanDropInsert(  
    int insertionIndex  
)
```

### Parameters

*insertionIndex*

Type: **SystemInt32**

Zero-based index of insertion.

### Return Value

Type: **Boolean**

A value of **true** if item can be dropped; otherwise **false**.

## ► Remarks

This method is always called whilst drawing an editor GUI.

## ► See Also

## Reference

[IReorderableListDropTarget Interface](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# IReorderableListDropTargetProcess Method

Processes the current drop insertion operation when [CanDropInsert\(Int32\)](#) returns a value of `true` to process, accept or cancel.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
void ProcessDropInsertion(  
    int insertionIndex  
)
```

### Parameters

*insertionIndex*

Type: **SystemInt32**

Zero-based index of insertion.

## ► Remarks

This method is always called whilst drawing an editor GUI.

This method is only called when [CanDropInsert\(Int32\)](#) returns a value of `true`.

## ► See Also

Reference

**IReorderableListDropTarget Interface**  
Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ItemInsertedEventArgs Class

Arguments which are passed to [ItemInsertedEventHandler](#).

## ► Inheritance Hierarchy

**SystemObject SystemEventArgs**  
Rotorz.ReorderableListItemInsertedEventArgs

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public sealed class ItemInsertedEventArgs : Event
```

The [ItemInsertedEventArgs](#) type exposes the following members.

## ► Constructors

	Name	Description
≡	<a href="#">ItemInsertedEventArgs</a>	Initializes a new instance of <a href="#">ItemInsertedEventArgs</a> .

[Top](#)

## ► Properties

	Name	Description
≡	<a href="#">Adaptor</a>	Gets adaptor to reorderable list

container which contains element.

---

	<a href="#">ItemIndex</a>	Gets zero-based index of item which was inserted.
	<a href="#">WasDuplicated</a>	Indicates if inserted item was duplicated from another item.

---

[Top](#)

## ▲ See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ItemInsertedEventArgs Constructor

Initializes a new instance of [ItemInsertedEventArgs](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public ItemInsertedEventArgs(  
    IReorderableListAdapter adaptor,  
    int itemIndex,  
    bool wasDuplicated  
)
```

## Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)  
Reorderable list adaptor.

*itemIndex*

Type: **SystemInt32**  
Zero-based index of item.

*wasDuplicated*

Type: **SystemBoolean**  
Indicates if inserted item was duplicated from another item.

## ► See Also

[Reference](#)

**ItemInsertedEventArgs Class**  
**Rotorz.ReorderableList Namespace**

---

©2013-2015 Rotorz Limited. All rights reserved.

# ItemInsertedEventArgs Properties

The [ItemInsertedEventArgs](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Adaptor</a>	Gets adaptor to reorderable list container which contains element.
	<a href="#">ItemIndex</a>	Gets zero-based index of item which was inserted.
	<a href="#">WasDuplicated</a>	Indicates if inserted item was duplicated from another item.

[Top](#)

## See Also

### Reference

[ItemInsertedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemInsertedEventArgsAdaptor Property

Gets adaptor to reorderable list container which contains element.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public IReorderableListAdapter Adaptor { get; }
```

Property Value

Type: [IReorderableListAdapter](#)

## ► See Also

Reference

[ItemInsertedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemInsertedEventArgs.ItemIndex Property

Gets zero-based index of item which was inserted.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public int ItemIndex { get; }
```

Property Value

Type: [Int32](#)

## ► See Also

Reference

[ItemInsertedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemInsertedEventArgsWasDuplicate Property

Indicates if inserted item was duplicated from another item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public bool WasDuplicated { get; }
```

Property Value

Type: **Boolean**

## ► See Also

Reference

[ItemInsertedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemInsertedEventHandler Delegate

An event handler which is invoked after new list item is inserted.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public delegate void ItemInsertedEventHandler(  
    Object sender,  
    ItemInsertedEventArgs args  
)
```

### Parameters

*sender*

Type: **SystemObject**

Object which raised event.

*args*

Type: [Rotorz.ReorderableListItemInsertedEventArgs](#)

Event arguments.

## ► See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

# ItemMovedEventArgs Class

Arguments which are passed to [ItemMovedEventHandler](#).

## ► Inheritance Hierarchy

**SystemObject SystemEventArgs**  
Rotorz.ReorderableListEventArgs

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public sealed class ItemMovedEventArgs : EventArgs
```

The [ItemMovedEventArgs](#) type exposes the following members.

## ► Constructors

	Name	Description
≡	<a href="#">ItemMovedEventArgs</a>	Initializes a new instance of <a href="#">ItemMovedEventArgs</a> .

[Top](#)

## ► Properties

	Name	Description
≡	<a href="#">Adaptor</a>	Gets adaptor to reorderable list

container which contains element.



#### [NewItemIndex](#)

Gets new zero-based index of the item which was moved.



#### [OldItemIndex](#)

Gets old zero-based index of the item which was moved.

---

[Top](#)

## ► See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ItemMovedEventArgs Constructor

Initializes a new instance of [ItemMovedEventArgs](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public ItemMovedEventArgs(  
    IReorderableListAdapter adaptor,  
    int oldItemIndex,  
    int newItemIndex  
)
```

## Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

*oldItemIndex*

Type: [System.Int32](#)

Old zero-based index of item.

*newItemIndex*

Type: [System.Int32](#)

New zero-based index of item.

## ► See Also

[Reference](#)

[ItemMovedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# ItemMovedEventArgs Properties

The [ItemMovedEventArgs](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Adaptor</a>	Gets adaptor to reorderable list container which contains element.
	<a href="#">NewItemIndex</a>	Gets new zero-based index of the item which was moved.
	<a href="#">OldItemIndex</a>	Gets old zero-based index of the item which was moved.

[Top](#)

## See Also

[Reference](#)

[ItemMovedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemMovedEventArgsAdaptor Property

Gets adaptor to reorderable list container which contains element.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public IReorderableListAdapter Adaptor { get; }
```

Property Value

Type: [IReorderableListAdapter](#)

## ► See Also

Reference

[ItemMovedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemMovedEventArgsNewItemIndex Property

Gets new zero-based index of the item which was moved.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public int NewItemIndex { get; }
```

Property Value

Type: [Int32](#)

## ► See Also

Reference

[ItemMovedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemMovedEventArgsOldItemIndex Property

Gets old zero-based index of the item which was moved.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public int OldItemIndex { get; }
```

Property Value

Type: [Int32](#)

## ► See Also

Reference

[ItemMovedEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemMovedEventHandler Delegate

An event handler which is invoked after a list item is moved.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public delegate void ItemMovedEventHandler(  
    Object sender,  
    ItemMovedEventArgs args  
)
```

### Parameters

*sender*

Type: **SystemObject**

Object which raised event.

*args*

Type: [Rotorz.ReorderableListItemMovedEventArgs](#)

Event arguments.

## ► See Also

Reference

[Rotorz.ReorderableList Namespace](#)

# ItemMovingEventArgs Class

Arguments which are passed to [ItemMovingEventHandler](#).

## ► Inheritance Hierarchy

```
SystemObject SystemEventArgs
  System.ComponentModel.CancelEventArgs
    Rotorz.ReorderableList.ItemMovingEventArgs
```

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)  
Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public sealed class ItemMovingEventArgs : CancelE
```

The [ItemMovingEventArgs](#) type exposes the following members.

## ► Constructors

	Name	Description
≡	<a href="#">ItemMovingEventArgs</a>	Initializes a new instance of <a href="#">ItemMovingEventArgs</a> .

[Top](#)

## ► Properties

	Name	Description
--	------	-------------



## Adaptor

Gets adaptor to reorderable list container which contains element.



## Cancel

Gets or sets a value indicating whether the event should be canceled.  
(Inherited from **CancelEventArgs**.)



## DestinationItemIndex

Gets the new candidate zero-based index for the item.



## ItemIndex

Gets current zero-based index of item which is going to be moved.



## NewItemIndex

Gets zero-based index of item **after** it has been moved.

[Top](#)

## See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# ItemMovingEventArgs Constructor

Initializes a new instance of [ItemMovingEventArgs](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public ItemMovingEventArgs(  
    IReorderableListAdapter adaptor,  
    int itemIndex,  
    int destinationItemIndex  
)
```

## Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

*itemIndex*

Type: [System.Int32](#)

Zero-based index of item.

*destinationItemIndex*

Type: [System.Int32](#)

Xero-based index of item destination.

## ► See Also

[Reference](#)

[ItemMovingEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# ItemMovingEventArgs Properties

The [ItemMovingEventArgs](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Adaptor</a>	Gets adaptor to reorderable list container which contains element.
	<a href="#">Cancel</a>	Gets or sets a value indicating whether the event should be canceled. (Inherited from <a href="#">CancelEventArgs</a> .)
	<a href="#">DestinationItemIndex</a>	Gets the new candidate zero-based index for the item.
	<a href="#">ItemIndex</a>	Gets current zero-based index of item which is going to be moved.
	<a href="#">NewItemIndex</a>	Gets zero-based index of item <b>after</b> it has been moved.

[Top](#)

## See Also

[Reference](#)

[ItemMovingEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# ItemMovingEventArgsAdaptor Property

Gets adaptor to reorderable list container which contains element.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public IReorderableListAdapter Adaptor { get; }
```

Property Value

Type: [IReorderableListAdapter](#)

## ► See Also

Reference

[ItemMovingEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemMovingEventArgsDestinationIndex Property

Gets the new candidate zero-based index for the item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public int DestinationIndex { get; }
```

Property Value

Type: [Int32](#)

## ► See Also

Reference

[ItemMovingEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

[ItemMovingEventArgsNewItemIndex](#)

# ItemMovingEventArgs.ItemIndex Property

Gets current zero-based index of item which is going to be moved.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public int ItemIndex { get; }
```

Property Value

Type: [Int32](#)

## ► See Also

Reference

[ItemMovingEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemMovingEventArgsNewItemIndex Property

Gets zero-based index of item **after** it has been moved.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public int NewItemIndex { get; }
```

Property Value

Type: [Int32](#)

## ► See Also

Reference

[ItemMovingEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

[ItemMovingEventArgsDestinationItemIndex](#)

# ItemMovingEventHandler Delegate

An event handler which is invoked before a list item is moved.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public delegate void ItemMovingEventHandler(  
    Object sender,  
    ItemMovingEventArgs args  
)
```

### Parameters

*sender*

Type: [SystemObject](#)

Object which raised event.

*args*

Type: [Rotorz.ReorderableListItemMovingEventArgs](#)

Event arguments.

## ► Remarks

Moving of item can be cancelled by setting **Cancel** to `true`.

## ► See Also

Reference

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ItemRemovingEventArgs Class

Arguments which are passed to [ItemRemovingEventHandler](#).

## ► Inheritance Hierarchy

```
SystemObject SystemEventArgs
  System.ComponentModel.CancelEventArgs
    Rotorz.ReorderableList.ItemRemovingEventArgs
```

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)  
Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public sealed class ItemRemovingEventArgs : Cancel
```

The [ItemRemovingEventArgs](#) type exposes the following members.

## ► Constructors

	Name	Description
≡	<a href="#">ItemRemovingEventArgs</a>	Initializes a new instance of <a href="#">ItemRemovingEventArgs</a> .

[Top](#)

## ► Properties

Name	Description
------	-------------

---

	<a href="#">Adaptor</a>	Gets adaptor to reorderable list container which contains element.
	<a href="#">Cancel</a>	Gets or sets a value indicating whether the event should be canceled. (Inherited from <a href="#">CancelEventArgs</a> .)
	<a href="#">ItemIndex</a>	Gets zero-based index of item which is being removed.

---

[Top](#)

## ▲ See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ItemRemovingEventArgs Constructor

Initializes a new instance of [ItemRemovingEventArgs](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public ItemRemovingEventArgs(  
    IReorderableListAdapter adaptor,  
    int itemIndex  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)  
Reorderable list adaptor.

*itemIndex*

Type: [System.Int32](#)  
Zero-based index of item.

## ► See Also

[Reference](#)

[ItemRemovingEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)



# ItemRemovingEventArgs Properties

The [ItemRemovingEventArgs](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Adaptor</a>	Gets adaptor to reorderable list container which contains element.
	<a href="#">Cancel</a>	Gets or sets a value indicating whether the event should be canceled. (Inherited from <a href="#">CancelEventArgs</a> .)
	<a href="#">ItemIndex</a>	Gets zero-based index of item which is being removed.

[Top](#)

## See Also

[Reference](#)

[ItemRemovingEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemRemovingEventArgsAdaptor Property

Gets adaptor to reorderable list container which contains element.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public IReorderableListAdapter Adaptor { get; }
```

Property Value

Type: [IReorderableListAdapter](#)

## ► See Also

Reference

[ItemRemovingEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemRemovingEventArgs.ItemIndex Property

Gets zero-based index of item which is being removed.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public int ItemIndex { get; }
```

Property Value

Type: [Int32](#)

## ► See Also

Reference

[ItemRemovingEventArgs Class](#)

[Rotorz.ReorderableList Namespace](#)

# ItemRemovingEventHandler Delegate

An event handler which is invoked before a list item is removed.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public delegate void ItemRemovingEventHandler(  
    Object sender,  
    ItemRemovingEventArgs args  
)
```

### Parameters

*sender*

Type: **SystemObject**

Object which raised event.

*args*

Type: [Rotorz.ReorderableListItemRemovingEventArgs](#)

Event arguments.

## ► Remarks

Item removal can be cancelled by setting **Cancel** to **true**.

## ► See Also

Reference

## Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControl Class

Base class for custom reorderable list control.

## ► Inheritance Hierarchy

**SystemObject** Rotorz.ReorderableListReorderableListControl

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public class ReorderableListControl
```

The **ReorderableListControl** type exposes the following members.

## ► Constructors

	Name	Description
≡	<a href="#">ReorderableListControl</a>	Initializes a instance of Reorderabl
≡	<a href="#">ReorderableListControl(ReorderableListFlags)</a>	Initializes a instance of Reorderabl

[Top](#)

## ► Methods

Name	Description
.AddItem	Add item to end of list and raise the even ItemInse
AddItemsToMenu	Invoked to generate context menu for item.
CalculateListHeight(IReorderableListAdapter)	Calculate height of control in pixels.
CalculateListHeight(Int32, Single)	Calculate height of control in pixels.
ClearAll	Remove items from list.
DoCommand(String, Int32, IReorderableListAdapter)	Call to manually perform command.
DoCommand(GUIContent, Int32, IReorderableListAdapter)	Call to manually perform command.
Draw(IReorderableListAdapter)	Draw layout.

		version control.
≡	Draw(Rect, IReorderableListAdapter)	Draw list control w absolute position
≡	Draw(IReorderableListAdapter, ReorderableListControlDrawEmpty)	Draw layout version control.
≡	Draw(Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAbsolute)	Draw list control w absolute position
≡ S	DrawControlFromState(IReorderableListAdapter, ReorderableListControlDrawEmpty, ReorderableListFlags)	Generate and draw control fr state obj
≡ S	DrawControlFromState(Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)	Generate and draw control fr state obj
💡	DrawDropIndicator	Draws drop insertion indicator
💡	DuplicateItem	Duplicates specified item and raises the event ItemInse
💡	HandleCommand	Invoked

handle  
context  
command

💡	<a href="#">InsertItem</a>	Insert item at specific index and raises the event <a href="#">ItemInserted</a> .
💡	<a href="#">MoveItem</a>	Move item from source index to destination index.
💡	<a href="#">OnAddMenuClicked</a>	Raises event when add menu button is clicked.
💡	<a href="#">OnItemInserted</a>	Raises event after list is inserted or duplicate.
💡	<a href="#">OnItemMoved</a>	Raises event after list has been moved.
💡	<a href="#">OnItemMoving</a>	Raises event immediately before list item is moved and provides opportunity.

to cancel

---

💡	<a href="#">OnItemRemoving</a>	Raises event before list item is removed. Provides opportunity to cancel.
💡	<a href="#">RemoveItem</a>	Removes specified item.

---

[Top](#)

## Fields

	Name	Description
💡 S	<a href="#">AnchorBackgroundColor</a>	Background color of anchor list item.
💡 S	<a href="#">CommandClearAll</a>	Content for "Clear All" command.
💡 S	<a href="#">CommandDuplicate</a>	Content for "Duplicate" command.
💡 S	<a href="#">CommandInsertAbove</a>	Content for "Insert Above" command.
💡 S	<a href="#">CommandInsertBelow</a>	Content for "Insert Below" command.
💡 S	<a href="#">CommandMoveToBottom</a>	Content for "Move to Bottom" command.
💡 S	<a href="#">CommandMoveToTop</a>	Content for "Move to Top"

command.

 	<a href="#">CommandRemove</a>	Content for "Remove" command.
  	<a href="#">DefaultContextHandler</a>	Default functionality to handle context command.
 	<a href="#">TargetBackgroundColor</a>	Background color of target slot when dragging list item.

[Top](#)

## Properties

	Name	Description
 	<a href="#">ContainerStyle</a>	Gets or sets style used to draw background of list control.
 	<a href="#">CurrentItemTotalPosition</a>	Gets the total position of the list item that is currently being drawn.
 	<a href="#">CurrentListControlID</a>	Gets the control ID of the list that is currently being drawn.
 	<a href="#">CurrentListPosition</a>	Gets the position of the list control that is currently being drawn.
 	<a href="#">Flags</a>	Gets or sets flags which affect behavior of control.
 	<a href="#">FooterButtonStyle</a>	Gets or sets style used to draw footer buttons.

	<a href="#">HorizontalLineAtEnd</a>	Gets or sets a boolean value indicating whether a horizontal line should be shown below the last list item at the end of the list control.
	<a href="#">HorizontalLineAtStart</a>	Gets or sets a boolean value indicating whether a horizontal line should be shown above the first list item at the start of the list control.
	<a href="#">HorizontalLineColor</a>	Gets or sets the color of the horizontal lines that appear between list items.
	<a href="#">ItemButtonStyle</a>	Gets or sets style used to draw list item buttons (like the remove button).

[Top](#)

## Events

	Name	Description
	<a href="#">AddMenuClicked</a>	Occurs when add menu button is clicked.
	<a href="#">ItemInserted</a>	Occurs after list item is inserted or duplicated.
	<a href="#">ItemMoved</a>	Occurs after list item has been moved.
	<a href="#">ItemMoving</a>	Occurs immediately before list item is moved allowing for move

operation to be cancelled.



### [ItemRemoving](#)

Occurs before list item is removed and allowing for remove operation to be cancelled.

[Top](#)

## ► See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControl Constructor

## ► Overload List

Name	Description
 <a href="#">ReorderableListControl</a>	Initializes a instance of Reorderabl
 <a href="#">ReorderableListControl(ReorderableListFlags)</a>	Initializes a instance of Reorderabl

[Top](#)

## ► See Also

### Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControl Constructor

Initializes a new instance of [ReorderableListControl](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public ReorderableListControl()
```

## ► See Also

Reference

[ReorderableListControl Class](#)

[ReorderableListControl Overload](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControl Constructor (ReorderableListFlags)

Initializes a new instance of [ReorderableListControl](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public ReorderableListControl(  
    ReorderableListFlags flags  
)
```

### Parameters

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)  
Optional flags which affect behavior of control.

## ► See Also

### Reference

[ReorderableListControl Class](#)

[ReorderableListControl Overload](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControl Fields

The [ReorderableListControl](#) type exposes the following members.

## Fields

	Name	Description
• S	<a href="#">AnchorBackgroundColor</a>	Background color of anchor list item.
• S	<a href="#">CommandClearAll</a>	Content for "Clear All" command.
• S	<a href="#">CommandDuplicate</a>	Content for "Duplicate" command.
• S	<a href="#">CommandInsertAbove</a>	Content for "Insert Above" command.
• S	<a href="#">CommandInsertBelow</a>	Content for "Insert Below" command.
• S	<a href="#">CommandMoveToBottom</a>	Content for "Move to Bottom" command.
• S	<a href="#">CommandMoveToTop</a>	Content for "Move to Top" command.
• S	<a href="#">CommandRemove</a>	Content for "Remove" command.
• S E	<a href="#">DefaultContextHandler</a>	Default functionality to handle context command.
• S	<a href="#">TargetBackgroundColor</a>	Background color of target

slot when dragging list item.

---

[Top](#)

## ◀ See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlAnchorBackgroundField

Background color of anchor list item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static readonly Color AnchorBackgroundColor
```

Field Value

Type: [Color](#)

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlCommandClearField

Content for "Clear All" command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected static readonly GUIContent CommandClear
```

Field Value

Type: [GUIContent](#)

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlCommandDuplicateField

Content for "Duplicate" command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    [UnityScript](#)

[Copy](#)

```
protected static readonly GUIContent CommandDuplicate
```

Field Value

Type: [GUIContent](#)

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlCommandInser Field

Content for "Insert Above" command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected static readonly GUIContent CommandInser
```

Field Value

Type: [GUIContent](#)

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlCommandInser Field

Content for "Insert Below" command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected static readonly GUIContent CommandInser
```

Field Value

Type: [GUIContent](#)

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlCommandMoveField

Content for "Move to Bottom" command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    [UnityScript](#)

[Copy](#)

```
protected static readonly GUIContent CommandMove1
```

Field Value

Type: [GUIContent](#)

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlCommandMoveField

Content for "Move to Top" command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected static readonly GUIContent CommandMove1
```

Field Value

Type: [GUIContent](#)

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlCommandRemoveField

Content for "Remove" command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    [UnityScript](#)

[Copy](#)

```
protected static readonly GUIContent CommandRemoveField;
```

Field Value

Type: [GUIContent](#)

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlDefaultContentField

Default functionality to handle context command.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
protected static readonly MenuFunction2 DefaultCo
```

Field Value

Type: **MenuFunction2**

## ► Examples

Can be used when adding custom items to the context menu:

C#    UnityScript

Copy

```
protected override void AddItemsToMenu(GenericMer  
    var specialCommand = new GUIContent("Special  
    menu.AddItem(specialCommand, false, defaultCo  
}
```

## ► See Also

Reference

[ReorderableListControl Class](#)

Rotorz.ReorderableList Namespace  
ReorderableListControlAddItemsToMenu(GenericMenu, Int32,  
IReorderableListAdapter)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlTargetBackgroundField

Background color of target slot when dragging list item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static readonly Color TargetBackgroundColor
```

Field Value

Type: [Color](#)

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControl Methods

The [ReorderableListControl](#) type exposes the following members.

## Methods

Name	Description
<a href="#">AddItem</a>	Add item to end of list and raise the even <a href="#">ItemInserted</a> .
<a href="#">AddItemsToMenu</a>	Invoked to generate context menu for item.
<a href="#">CalculateListHeight(IReorderableListAdapter)</a>	Calculate height of control in pixels.
<a href="#">CalculateListHeight(Int32, Single)</a>	Calculate height of control in pixels.
<a href="#">ClearAll</a>	Remove items from list.
<a href="#">DoCommand(String, Int32, IReorderableListAdapter)</a>	Call to manually

		perform command
≡	<a href="#">DoCommand(GUIContent, Int32, IReorderableListAdapter)</a>	Call to manually perform command
≡	<a href="#">Draw(IReorderableListAdapter)</a>	Draw layout version control.
≡	<a href="#">Draw(Rect, IReorderableListAdapter)</a>	Draw list control w absolute position
≡	<a href="#">Draw(IReorderableListAdapter, ReorderableListControlDrawEmpty)</a>	Draw layout version control.
≡	<a href="#">Draw(Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAbsolute)</a>	Draw list control w absolute position
≡ S	<a href="#">DrawControlFromState(IReorderableListAdapter, ReorderableListControlDrawEmpty, ReorderableListFlags)</a>	Generate and draw control fr state obj
≡ S	<a href="#">DrawControlFromState(Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)</a>	Generate and draw control fr state obj
≡	<a href="#">DrawDropIndicator</a>	Draws drop insertion indicator

💡	DuplicateItem	Duplicate specific item and raises the event <a href="#">ItemInse</a>
💡	HandleCommand	Invoked handle context command
💡	InsertItem	Insert item at specified index and raises the event <a href="#">ItemInse</a>
💡	MoveItem	Move item from source index to destination index.
💡	OnAddMenuClicked	Raises event when add menu button is clicked
💡	OnItemInserted	Raises event after list is inserted or duplicate
💡	OnItemMoved	Raises event after list has been moved.



### [OnItemMoving](#)

Raises an event immediately before an item is moved and provides an opportunity to cancel.



### [OnItemRemoving](#)

Raises an event before an item is removed and provides an opportunity to cancel.



### [RemoveItem](#)

Removes a specified item.

---

[Top](#)

## See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControl.AddItem Method

Add item at end of list and raises the event [ItemInserted](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected void AddItem(  
    IReorderableListAdapter adaptor  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControl.AddItemstoMenu Method

Invoked to generate context menu for list item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#

UnityScript

Copy

```
protected virtual void AddItemsToMenu(  
    GenericMenu menu,  
    int itemIndex,  
    IReorderableListAdapter adaptor  
)
```

## Parameters

*menu*

Type: **GenericMenu**

Menu which can be populated.

*itemIndex*

Type: **SystemInt32**

Zero-based index of item which was right-clicked.

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

## ◀ See Also

Reference

ReorderableListControl Class  
Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlCalculateList Method

## ► Overload List

Name	Description
<a href="#">CalculateListHeight(IReorderableListAdapter)</a>	Calculate height of list control in pixels.
<a href="#">CalculateListHeight(Int32, Single)</a>	Calculate height of list control in pixels.

[Top](#)

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlCalculateList Method (IReorderableListAdapter)

Calculate height of list control in pixels.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public float CalculateListHeight(  
    IReorderableListAdapter adaptor  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

### Return Value

Type: [Single](#)

Required list height in pixels.

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[CalculateListHeight Overload](#)

[Rotorz.ReorderableList Namespace](#)



# ReorderableListControlCalculateList Method (Int32, Single)

Calculate height of list control in pixels.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public float CalculateListHeight(  
    int itemCount,  
    float itemHeight  
)
```

### Parameters

*itemCount*

Type: **SystemInt32**

Count of items in list.

*itemHeight*

Type: **SystemSingle**

Fixed height of list item.

### Return Value

Type: **Single**

Required list height in pixels.

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

**CalculateListHeight Overload**  
**Rotorz.ReorderableList Namespace**

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlClearAll Method

Remove all items from list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected bool ClearAll(  
    IReorderableListAdapter adaptor  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)  
Reorderable list adaptor.

### Return Value

Type: **Boolean**

Returns a value of [false](#) if operation was cancelled.

## ► Remarks

The event [ItemRemoving](#) is raised for each item prior to clearing array and allows entire operation to be cancelled.

## ► See Also

## Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlDoCommand Method

## ► Overload List

	Name	Description
≡	<a href="#">DoCommand(String, Int32, IReorderableListAdapter)</a>	Call to manually perform command.
≡	<a href="#">DoCommand(GUIContent, Int32, IReorderableListAdapter)</a>	Call to manually perform command.

[Top](#)

## ► See Also

### Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlDoCommand Method (String, Int32, IReorderableListAdapter)

Call to manually perform command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public bool DoCommand(  
    string commandName,  
    int itemIndex,  
    IReorderableListAdapter adaptor  
)
```

## Parameters

*commandName*

Type: [SystemString](#)

Name of command. This is the text shown in the context menu.

*itemIndex*

Type: [SystemInt32](#)

Zero-based index of item which was right-clicked.

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

## Return Value

Type: [Boolean](#)

A value of `true` if command was known; otherwise `false`.

## Remarks

Warning message is logged to console if attempted to execute unknown command.

## See Also

Reference

[ReorderableListControl Class](#)

[DoCommand Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlDoCommand Method (GUIContent, Int32, IReorderableListAdapter)

Call to manually perform command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public bool DoCommand(  
    GUIContent command,  
    int itemIndex,  
    IReorderableListAdapter adaptor  
)
```

## Parameters

*command*

Type: [GUIContent](#)

Content representing command.

*itemIndex*

Type: [SystemInt32](#)

Zero-based index of item which was right-clicked.

*adaptor*

Type: [Rotorz.ReorderableListIReorderableListAdapter](#)

Reorderable list adaptor.

## Return Value

Type: [Boolean](#)

A value of `true` if command was known; otherwise `false`.

## Remarks

Warning message is logged to console if attempted to execute unknown command.

## See Also

Reference

[ReorderableListControl Class](#)

[DoCommand Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlDraw Method

## ► Overload List

Name	Description
 <a href="#">Draw(IReorderableListAdapter)</a>	Draw layout version of list control.
 <a href="#">Draw(Rect, IReorderableListAdapter)</a>	Draw list control with absolute positioning.
 <a href="#">Draw(IReorderableListAdapter, ReorderableListControlDrawEmpty)</a>	Draw layout version of list control.
 <a href="#">Draw(Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAbsolute)</a>	Draw list control with absolute positioning.

[Top](#)

## ► See Also

### Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)



# ReorderableListControlDraw Method (IReorderableListAdapter)

Draw layout version of list control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public void Draw(  
    IReorderableListAdapter adaptor  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

## ► See Also

Reference

[ReorderableListControl Class](#)

[Draw Overload](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlDraw Method (Rect, IReorderableListAdapter)

Draw list control with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public void Draw(  
    Rect position,  
    IReorderableListAdapter adaptor  
)
```

### Parameters

*position*

Type: [Rect](#)

Position of list control in GUI.

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Draw Overload](#)

## Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlDraw Method (IReorderableListAdapter, ReorderableListControlDrawEmpty)

Draw layout version of list control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public void Draw(
    IReorderableListAdapter adaptor,
    ReorderableListControlDrawEmpty drawEmpty
)
```

## Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

*drawEmpty*

Type: [Rotorz.ReorderableList.ReorderableListControlDrawEmpty](#)

Delegate for drawing empty list.

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Draw Overload](#)

## Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlDraw Method

## IReorderableListAdapter,

## ReorderableListControlDrawEmptyA

Draw list control with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public void Draw(  
    Rect position,  
    IReorderableListAdapter adaptor,  
    ReorderableListControlDrawEmptyAbsolute drawEmpty)  
)
```

## Parameters

*position*

Type: **Rect**

Position of list control in GUI.

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

*drawEmpty*

Type: [Rotorz.ReorderableList.ReorderableListControlDrawEmptyA](#)

Delegate for drawing empty list.

## ▲ See Also

### Reference

[ReorderableListControl Class](#)

[Draw Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlDrawControl Method

## ► Overload List

Name	Description
  <a href="#">DrawControlFromState(IReorderableListAdapter, ReorderableListControlDrawEmpty, ReorderableListFlags)</a>	Generate and draw control for state obj
  <a href="#">DrawControlFromState(Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)</a>	Generate and draw control for state obj

[Top](#)

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlDrawControl Method (IReorderableListAdapter, ReorderableListControlDrawEmpty, ReorderableListFlags)

Generate and draw control from state object.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#

UnityScript

Copy

```
public static void DrawControlFromState(  
    IReorderableListAdapter adaptor,  
    ReorderableListControlDrawEmpty drawEmpty,  
    ReorderableListFlags flags  
)  
{  
    1  
}
```

## Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)  
Reorderable list adaptor.

*drawEmpty*

Type: [Rotorz.ReorderableList.ReorderableListControlDrawEmpty](#)  
Delegate for drawing empty list.

*flags*

Type: [Rotorz.ReorderableList.ReorderableListFlags](#)  
Optional flags to pass into list field.

## ▲ See Also

### Reference

[ReorderableListControl Class](#)

[DrawControlFromState Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlDrawControl Method (Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAction, ReorderableListFlags)

Generate and draw control from state object.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#

UnityScript

[Copy](#)

```
public static void DrawControlFromState(  
    Rect position,  
    IReorderableListAdapter adaptor,  
    ReorderableListControlDrawEmptyAbsolute drawEmpty,  
    ReorderableListFlags flags  
)
```

## Parameters

*position*

Type: [Rect](#)

Position of control.

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

*drawEmpty*

Type: [Rotorz.ReorderableList.ReorderableListControlDrawEmptyAction](#)

Delegate for drawing empty list.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[DrawControlFromState Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlDrawDropIndicator Method

Draws drop insertion indicator.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected virtual void DrawDropIndicator(  
    Rect position  
)
```

### Parameters

*position*

Type: **Rect**

Position if the drop indicator.

## ► Remarks

This method is only ever called during repaint events.

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)



# ReorderableListControlDuplicateItem Method

Duplicate specified item and raises the event [ItemInserted](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#    UnityScript

[Copy](#)

```
protected void DuplicateItem(  
    IReorderableListAdapter adaptor,  
    int itemIndex  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)  
Reorderable list adaptor.

*itemIndex*

Type: [System.Int32](#)  
Zero-based index of item.

## ◀ See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)



# ReorderableListControlHandleCommand

Invoked to handle context command.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#    UnityScript

[Copy](#)

```
protected virtual bool HandleCommand(  
    string commandName,  
    int itemIndex,  
    IReorderableListAdapter adaptor  
)
```

### Parameters

*commandName*

Type: **SystemString**

Name of command. This is the text shown in the context menu.

*itemIndex*

Type: **SystemInt32**

Zero-based index of item which was right-clicked.

*adaptor*

Type: [Rotorz.ReorderableListIReorderableListAdapter](#)

Reorderable list adaptor.

### Return Value

Type: **Boolean**

A value of **true** if command was known; otherwise **false**.

## ► Remarks

It is important to set the value of `GUI.changed` to `true` if any changes are made by command handler.

Default command handling functionality can be inherited:

C#    UnityScript

[Copy](#)

```
protected override bool HandleCommand(string commandName)
{
    if (base.HandleCommand(itemIndex, adaptor))
        return true;

    // Place custom command handling code here...
    switch (commandName) {
        case "Your Command":
            return true;
    }

    return false;
}
```

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControl.InsertItem Method

Insert item at specified index and raises the event [ItemInserted](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected void InsertItem(  
    IReorderableListAdapter adaptor,  
    int itemIndex  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)  
Reorderable list adaptor.

*itemIndex*

Type: [System.Int32](#)  
Zero-based index of item.

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)



# ReorderableListControlMoveItem Method

Move item from source index to destination index.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected void MoveItem(  
    IReorderableListAdapter adaptor,  
    int sourceIndex,  
    int destIndex  
)
```

## Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)  
Reorderable list adaptor.

*sourceIndex*

Type: **SystemInt32**  
Zero-based index of source item.

*destIndex*

Type: **SystemInt32**  
Zero-based index of destination index.

## ► See Also

[Reference](#)

ReorderableListControl Class  
Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlOnAddMenu Method

Raises event when add menu button is clicked.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected virtual void OnAddMenuClicked(  
    AddMenuClickedEventArgs args  
)
```

### Parameters

*args*

Type: [Rotorz.ReorderableListAddMenuClickedEventArgs](#)  
Event arguments.

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlOnItemInsert Method

Raises event after list item is inserted or duplicated.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected virtual void OnItemInserted(  
    ItemInsertedEventArgs args  
)
```

### Parameters

*args*

Type: [Rotorz.ReorderableListItemInsertedEventArgs](#)  
Event arguments.

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControl.OnItemMove Method

Raises event after list item has been moved.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
protected virtual void OnItemMoved(  
    ItemMovedEventArgs args  
)
```

### Parameters

*args*

Type: [Rotorz.ReorderableList.ItemMovedEventArgs](#)  
Event arguments.

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControl.OnItemMovir Method

Raises event immediately before list item is moved and provides opportunity to cancel.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
protected virtual void OnItemMoving(  
    ItemMovingEventArgs args  
)
```

### Parameters

*args*

Type: [Rotorz.ReorderableList.ItemMovingEventArgs](#)  
Event arguments.

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControl.OnItemRemoving Method

Raises event before list item is removed and provides opportunity to cancel.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
protected virtual void OnItemRemoving(  
    ItemRemovingEventArgs args  
)
```

### Parameters

*args*

Type: [Rotorz.ReorderableList.ItemRemovingEventArgs](#)  
Event arguments.

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlRemoveItem Method

Remove specified item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#    UnityScript

[Copy](#)

```
protected bool RemoveItem(  
    IReorderableListAdapter adaptor,  
    int itemIndex  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)  
Reorderable list adaptor.

*itemIndex*

Type: [System.Int32](#)  
Zero-based index of item.

### Return Value

Type: **Boolean**

Returns a value of [false](#) if operation was cancelled.

## ◀ Remarks

The event [ItemRemoving](#) is raised prior to removing item and allows removal to be cancelled.

## ◀ See Also

### Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControl Properties

The [ReorderableListControl](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">ContainerStyle</a>	Gets or sets style used to draw background of list control.
	<a href="#">CurrentItemTotalPosition</a>	Gets the total position of the list item that is currently being drawn.
	<a href="#">CurrentListControlID</a>	Gets the control ID of the list that is currently being drawn.
	<a href="#">CurrentListPosition</a>	Gets the position of the list control that is currently being drawn.
	<a href="#">Flags</a>	Gets or sets flags which affect behavior of control.
	<a href="#">FooterButtonStyle</a>	Gets or sets style used to draw footer buttons.
	<a href="#">HorizontalLineAtEnd</a>	Gets or sets a boolean value indicating whether a horizontal line should be shown below the last list item at the end of the list control.

---

	<a href="#">HorizontalLineAtStart</a>	Gets or sets a boolean value indicating whether a horizontal line should be shown above the first list item at the start of the list control.
	<a href="#">HorizontalLineColor</a>	Gets or sets the color of the horizontal lines that appear between list items.
	<a href="#">ItemButtonStyle</a>	Gets or sets style used to draw list item buttons (like the remove button).

---

[Top](#)

## ◀ See Also

### Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlContainerStyle Property

Gets or sets style used to draw background of list control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public GUIStyle ContainerStyle { get; set; }
```

Property Value

Type: **GUIStyle**

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

[ReorderableListStylesContainer](#)

# ReorderableListControlCurrentItemT Property

Gets the total position of the list item that is currently being drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static Rect CurrentItemTotalPosition { get
```



Property Value

Type: [Rect](#)

## ► Remarks

The value of this property should be ignored for [Layout](#) type events when using reorderable list controls with automatic layout.

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlCurrentListControlID Property

Gets the control ID of the list that is currently being drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static int CurrentListControlID { get; }
```

Property Value

Type: [Int32](#)

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlCurrentListPosition Property

Gets the position of the list control that is currently being drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static Rect CurrentListPosition { get; }
```

Property Value

Type: **Rect**

## ► Remarks

The value of this property should be ignored for **Layout** type events when using reorderable list controls with automatic layout.

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlFlags Property

Gets or sets flags which affect behavior of control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public ReorderableListFlags Flags { get; set; }
```

Property Value

Type: [ReorderableListFlags](#)

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlFooterButton Property

Gets or sets style used to draw footer buttons.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public GUIStyle FooterButtonStyle { get; set; }
```

Property Value

Type: **GUIStyle**

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

[ReorderableListStylesFooterButton](#)

# ReorderableListControlHorizontalLineAtEnd Property

Gets or sets a boolean value indicating whether a horizontal line should be shown below the last list item at the end of the list control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public bool HorizontalLineAtEnd { get; set; }
```

Property Value

Type: **Boolean**

## ► Remarks

Horizontal line is not drawn for an empty list regardless of the value of this property.

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlHorizontalLineAtStart Property

Gets or sets a boolean value indicating whether a horizontal line should be shown above the first list item at the start of the list control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public bool HorizontalLineAtStart { get; set; }
```

Property Value

Type: **Boolean**

## ► Remarks

Horizontal line is not drawn for an empty list regardless of the value of this property.

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlHorizontalLineColor Property

Gets or sets the color of the horizontal lines that appear between list items.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public Color HorizontalLineColor { get; set; }
```

Property Value

Type: **Color**

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlItemButtonStyle Property

Gets or sets style used to draw list item buttons (like the remove button).

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public GUIStyle ItemButtonStyle { get; set; }
```

Property Value

Type: **GUIStyle**

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

[ReorderableListStylesItemButton](#)

# ReorderableListControl Events

The [ReorderableListControl](#) type exposes the following members.

## Events

	Name	Description
	<a href="#">AddMenuClicked</a>	Occurs when add menu button is clicked.
	<a href="#">ItemInserted</a>	Occurs after list item is inserted or duplicated.
	<a href="#">ItemMoved</a>	Occurs after list item has been moved.
	<a href="#">ItemMoving</a>	Occurs immediately before list item is moved allowing for move operation to be cancelled.
	<a href="#">ItemRemoving</a>	Occurs before list item is removed and allowing for remove operation to be cancelled.

[Top](#)

## See Also

### Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlAddMenuClickEvent

Occurs when add menu button is clicked.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public event AddMenuClickedEventHandler AddMenuC]
```

Value

Type: [Rotorz.ReorderableListAddMenuClickedEventHandler](#)

## ► Remarks

Add menu button is only shown when there is at least one subscriber to this event.

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlItemInserted Event

Occurs after list item is inserted or duplicated.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public event ItemInsertedEventHandler ItemInserted
```

Value

Type: [Rotorz.ReorderableListItemInsertedEventHandler](#)

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlItemMoved Event

Occurs after list item has been moved.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public event ItemMovedEventHandler ItemMoved
```

**Value**

Type: [Rotorz.ReorderableListItemMovedEventHandler](#)

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlItemMoving Event

Occurs immediately before list item is moved allowing for move operation to be cancelled.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public event ItemMovingEventHandler ItemMoving
```

Value

Type: [Rotorz.ReorderableListItemMovingEventHandler](#)

## ► See Also

[Reference](#)

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlItemRemovir Event

Occurs before list item is removed and allowing for remove operation to be cancelled.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public event ItemRemovingEventHandler ItemRemovir
```

Value

Type: [Rotorz.ReorderableListItemRemovingEventHandler](#)

## ► See Also

Reference

[ReorderableListControl Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlDrawEmptyDelegate

Invoked to draw content for empty list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public delegate void DrawEmpty()
```

## ► Remarks

Callback should make use of [GUILayout](#) to present controls.

## ► Examples

The following listing displays a label for empty list control:

C#    UnityScript

[Copy](#)

```
using Rotorz.ReorderableList;
using System.Collections.Generic;
using UnityEditor;
using UnityEngine;

public class ExampleWindow : EditorWindow {
    private List<string> _list;

    private void OnEnable() {
```

```
        _list = new List<string>();
    }
    private void OnGUI() {
        ReorderableListGUI.ListField(_list, ReorderableListGUI.CreateHeader("List"), ReorderableListGUI.CreateFooter("List"));
    }

    private string DrawEmptyMessage() {
        GUILayout.Label("List is empty!", EditorStyles.boldLabel);
    }
}
```



## See Also

### Reference

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListControlDrawEmptyA Delegate

Invoked to draw content for empty list with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public delegate void DrawEmptyAbsolute(  
    Rect position  
)
```

### Parameters

*position*

Type: **Rect**

Position of empty content.

## ► See Also

Reference

[Rotorz.ReorderableList Namespace](#)

# ReorderableListControlItemDrawer $T$ Delegate

Invoked to draw list item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#    UnityScript

[Copy](#)

```
public delegate T ItemDrawer<T>(  
    Rect position,  
    T item  
)
```

### Parameters

*position*

Type: **Rect**

Position of list item.

*item*

Type: **T**

The list item.

### Type Parameters

*T*

Type of item list.

### Return Value

Type: **T**

The modified value.

## ► Remarks

GUI controls must be positioned absolutely within the given rectangle since list items must be sized consistently.

## ► Examples

The following listing presents a text field for each list item:

C#    UnityScript

[Copy](#)

```
using Rotorz.ReorderableList;
using System.Collections.Generic;
using UnityEditor;
using UnityEngine;

public class ExampleWindow : EditorWindow {
    public List<string> wishlist = new List<string>();

    private void OnGUI() {
        ReorderableListGUI.ListField(wishlist, DrawList);
    }

    private string DrawListItem(Rect position, string value) {
        // Text fields do not like `null` values!
        if (value == null)
            value = "";
        return EditorGUI.TextField(position, value);
    }
}
```

## ► See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListFlags Enumeration

Additional flags which can be passed into reorderable list field.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ▲ Syntax

C#    UnityScript

[Copy](#)

```
[FlagsAttribute]
public enum ReorderableListFlags
```

## ▲ Members

Member name	Value	Description
DisableReordering	1	Hide grab handles and disable reordering of list items.
HideAddButton	2	Hide add button at base of control.
HideRemoveButtons	4	Hide remove buttons from list items.
DisableContextMenu	8	Do not display context menu upon right-

			clicking grab handle.
DisableDuplicateCommand	16		Hide "Duplicate" option from context menu.
DisableAutoFocus	32		Do not automatically focus first control of newly added items.
ShowIndices	64		Show zero-based index of array elements.
DisableClipping	128		Do not attempt to clip items which are out of view.

## Examples

Multiple flags can be specified if desired:

C#

Copy

```
var flags = ReorderableListFlags.HideAddButton |  
ReorderableListGUI.ListField(list, flags);
```

## See Also

## Reference

### [Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUI Class

Utility class for drawing reorderable lists.

## ► Inheritance Hierarchy

**SystemObject** Rotorz.ReorderableListReorderableListGUI

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static class ReorderableListGUI
```

The ReorderableListGUI type exposes the following members.

## ► Methods

Name	Description
  <a href="#">CalculateListFieldHeight(Int32)</a>	Calculates height of field for absolute position.
  <a href="#">CalculateListFieldHeight(SerializedProperty)</a>	Calculates height of field for absolute position.

≡ S	<a href="#">CalculateListFieldHeight(IReorderableListAdapter)</a>	Calculate height of field for adapter collection
≡ S	<a href="#">CalculateListFieldHeight(Int32, ReorderableListFlags)</a>	Calculate height of field for absolute position
≡ S	<a href="#">CalculateListFieldHeight(Int32, Single)</a>	Calculate height of field for absolute position
≡ S	<a href="#">CalculateListFieldHeight(SerializedProperty, ReorderableListFlags)</a>	Calculate height of field for absolute position
≡ S	<a href="#">CalculateListFieldHeight(IReorderableListAdapter, ReorderableListFlags)</a>	Calculate height of field for adapter collection
≡ S	<a href="#">CalculateListFieldHeight(Int32, Single, ReorderableListFlags)</a>	Calculate height of field for absolute position
≡ S	<a href="#">DefaultItemDrawerT</a>	Default drawer implementation

≡	■	ListField(SerializedProperty)	Draw li control serializ propert
≡	■	ListField(IReorderableListAdapter)	Draw li control adapte collecti
≡	■	ListField(SerializedProperty, ReorderableListControlDrawEmpty)	Draw li control serializ propert
≡	■	ListField(SerializedProperty, ReorderableListFlags)	Draw li control serializ propert
≡	■	ListField(SerializedProperty, Single)	Draw li control serializ propert
≡	■	ListField(IReorderableListAdapter, ReorderableListControlDrawEmpty)	Draw li control adapte collecti
≡	■	ListField(IReorderableListAdapter, ReorderableListFlags)	Draw li control adapte collecti
≡	■	ListField(SerializedProperty, ReorderableListControlDrawEmpty, ReorderableListFlags)	Draw li control serializ propert

≡	■	ListField(SerializedProperty, Single, ReorderableListControlDrawEmpty)	Draw li control serializ propert
≡	■	ListField(SerializedProperty, Single, ReorderableListFlags)	Draw li control serializ propert
≡	■	ListField(IReorderableListAdapter, ReorderableListControlDrawEmpty, ReorderableListFlags)	Draw li control adapte collecti
≡	■	ListField(SerializedProperty, Single, ReorderableListControlDrawEmpty, ReorderableListFlags)	Draw li control serializ propert
≡	■	ListFieldT(IListT, ReorderableListControlItemDrawerT)	Draw li control
≡	■	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmpty)	Draw li control
≡	■	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListFlags)	Draw li control
≡	■	ListFieldT(IListT, ReorderableListControlItemDrawerT, Single)	Draw li control
≡	■	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmpty, ReorderableListFlags)	Draw li control
≡	■		

	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmpty, Single)	Draw li control
≡  S	ListFieldT(IListT, ReorderableListControlItemDrawerT, Single, ReorderableListFlags)	Draw li control
≡  S	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmpty, Single, ReorderableListFlags)	Draw li control
≡  S	ListFieldAbsolute(Rect, IReorderableListAdaptor)	Draw li control adapte collecti
≡  S	ListFieldAbsolute(Rect, SerializedProperty)	Draw li control serializ propert
≡  S	ListFieldAbsolute(Rect, IReorderableListAdaptor, ReorderableListControlDrawEmptyAbsolute)	Draw li control adapte collecti
≡  S	ListFieldAbsolute(Rect, IReorderableListAdaptor, ReorderableListFlags)	Draw li control adapte collecti
≡  S	ListFieldAbsolute(Rect, SerializedProperty, ReorderableListControlDrawEmptyAbsolute)	Draw li control serializ propert
≡  S	ListFieldAbsolute(Rect, SerializedProperty,	Draw li

	<code>ReorderableListFlags)</code>	control serializ propert
≡ S	<code>ListFieldAbsolute(Rect, SerializedProperty, Single)</code>	Draw li control serializ propert
≡ S	<code>ListFieldAbsolute(Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)</code>	Draw li control adapte collecti
≡ S	<code>ListFieldAbsolute(Rect, SerializedProperty, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)</code>	Draw li control serializ propert
≡ S	<code>ListFieldAbsolute(Rect, SerializedProperty, Single, ReorderableListControlDrawEmptyAbsolute)</code>	Draw li control serializ propert
≡ S	<code>ListFieldAbsolute(Rect, SerializedProperty, Single, ReorderableListFlags)</code>	Draw li control serializ propert
≡ S	<code>ListFieldAbsolute(Rect, SerializedProperty, Single, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)</code>	Draw li control serializ propert
≡ S	<code>ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT)</code>	Draw li control absolut positio
≡ S		

	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute)	Draw li control absolut position
≡  S	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListFlags)	Draw li control absolut position
≡  S	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, Single)	Draw li control absolut position
≡  S	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)	Draw li control absolut position
≡  S	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute, Single)	Draw li control absolut position
≡  S	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, Single, ReorderableListFlags)	Draw li control absolut position
≡  S	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute, Single, ReorderableListFlags)	Draw li control absolut position
≡  S	TextFieldItemDrawer	Draws allowin items to edited.

  	Title(String)	Draw title control field.
  	Title(GUIContent)	Draw title control field.
 	Title(Rect, String)	Draw title control field with absolute position.
 	Title(Rect, GUIContent)	Draw title control field with absolute position.

[Top](#)

## Fields

	Name	Description
 	DefaultItemHeight	Default list item height is 18 pixels.

[Top](#)

## Properties

	Name	Description
 	CurrentItemIndex	Gets the zero-based index of the list item that is currently being drawn; or a value of -1 if no item is

currently being drawn.

 	<a href="#">CurrentItemTotalPosition</a>	Gets the total position of the list item that is currently being drawn.
 	<a href="#">CurrentListControlID</a>	Gets the control ID of the list that is currently being drawn.
 	<a href="#">CurrentListPosition</a>	Gets the position of the list control that is currently being drawn.
 	<a href="#">IndexOfChangedItem</a>	Gets or sets the zero-based index of the last item that was changed. A value of -1 indicates that no item was changed by list.

[Top](#)

## See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUI Fields

The [ReorderableListGUI](#) type exposes the following members.

## Fields

	Name	Description
•  	<a href="#">DefaultItemHeight</a>	Default list item height is 18 pixels.

[Top](#)

## See Also

[Reference](#)

[ReorderableListGUI Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUIDefaultItemHeightField

Default list item height is 18 pixels.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public const float DefaultItemHeight
```

Field Value

Type: **Single**

## ► See Also

Reference

[ReorderableListGUI Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUI Methods

The [ReorderableListGUI](#) type exposes the following members.

## Methods

Name	Description
<a href="#">CalculateListFieldHeight(Int32)</a>	Calculates height of field for absolute position
<a href="#">CalculateListFieldHeight(SerializedProperty)</a>	Calculates height of field for absolute position
<a href="#">CalculateListFieldHeight(IReorderableListAdapter)</a>	Calculates height of field for adapter collection
<a href="#">CalculateListFieldHeight(Int32, ReorderableListFlags)</a>	Calculates height of field for absolute position
<a href="#">CalculateListFieldHeight(Int32, Single)</a>	Calculates height of field for absolute position

	position	
≡ S	CalculateListFieldHeight(SerializedProperty, ReorderableListFlags)	Calculate height of field for absolute position
≡ S	CalculateListFieldHeight(IReorderableListAdaptor, ReorderableListFlags)	Calculate height of field for adapter collection
≡ S	CalculateListFieldHeight(Int32, Single, ReorderableListFlags)	Calculate height of field for absolute position
≡ S	DefaultItemDrawerT	Default drawer implementation
≡ S	ListField(SerializedProperty)	Draw list control serializing properties
≡ S	ListField(IReorderableListAdaptor)	Draw list control adapter collection
≡ S	ListField(SerializedProperty, ReorderableListControlDrawEmpty)	Draw list control serializing properties
≡ S		

	<code>ListField(SerializedProperty, ReorderableListFlags)</code>	Draw li control serializ propert
≡  	<code>ListField(SerializedProperty, Single)</code>	Draw li control serializ propert
≡  	<code>ListField(IReorderableListAdaptor, ReorderableListControlDrawEmpty)</code>	Draw li control adapte collecti
≡  	<code>ListField(IReorderableListAdaptor, ReorderableListFlags)</code>	Draw li control adapte collecti
≡   	<code>ListField(SerializedProperty, ReorderableListControlDrawEmpty, ReorderableListFlags)</code>	Draw li control serializ propert
≡  	<code>ListField(SerializedProperty, Single, ReorderableListControlDrawEmpty)</code>	Draw li control serializ propert
≡  	<code>ListField(SerializedProperty, Single, ReorderableListFlags)</code>	Draw li control serializ propert
≡  	<code>ListField(IReorderableListAdaptor, ReorderableListControlDrawEmpty, ReorderableListFlags)</code>	Draw li control adapte collecti

≡ ♦ S	ListField(SerializedProperty, Single, ReorderableListControlDrawEmpty, ReorderableListFlags)	Draw li control serializ propert
≡ ♦ S	ListFieldT(IListT, ReorderableListControlItemDrawerT)	Draw li control
≡ ♦ S	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmpty)	Draw li control
≡ ♦ S	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListFlags)	Draw li control
≡ ♦ S	ListFieldT(IListT, ReorderableListControlItemDrawerT, Single)	Draw li control
≡ ♦ S	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmpty, ReorderableListFlags)	Draw li control
≡ ♦ S	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmpty, Single)	Draw li control
≡ ♦ S	ListFieldT(IListT, ReorderableListControlItemDrawerT, Single, ReorderableListFlags)	Draw li control
≡ ♦ S	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmpty, Single, ReorderableListFlags)	Draw li control
≡ ♦ S	ListFieldAbsolute(Rect, IReorderableListAdapter)	Draw li control

		adapte collecti
≡	<a href="#">S ListFieldAbsolute(Rect, SerializedProperty)</a>	Draw li control serializ propert
≡	<a href="#">S ListFieldAbsolute(Rect, IReorderableListAdaptor, ReorderableListControlDrawEmptyAbsolute)</a>	Draw li control adapte collecti
≡	<a href="#">S ListFieldAbsolute(Rect, IReorderableListAdaptor, ReorderableListFlags)</a>	Draw li control adapte collecti
≡	<a href="#">S ListFieldAbsolute(Rect, SerializedProperty, ReorderableListControlDrawEmptyAbsolute)</a>	Draw li control serializ propert
≡	<a href="#">S ListFieldAbsolute(Rect, SerializedProperty, ReorderableListFlags)</a>	Draw li control serializ propert
≡	<a href="#">S ListFieldAbsolute(Rect, SerializedProperty, Single)</a>	Draw li control serializ propert
≡	<a href="#">S ListFieldAbsolute(Rect, IReorderableListAdaptor, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)</a>	Draw li control adapte collecti
≡	<a href="#">S ListFieldAbsolute(Rect, SerializedProperty,</a>	Draw li

	<code>ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)</code>	control serializ propert
≡ S	<code>ListFieldAbsolute(Rect, SerializedProperty, Single, ReorderableListControlDrawEmptyAbsolute)</code>	Draw li control serializ propert
≡ S	<code>ListFieldAbsolute(Rect, SerializedProperty, Single, ReorderableListFlags)</code>	Draw li control serializ propert
≡ S	<code>ListFieldAbsolute(Rect, SerializedProperty, Single, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)</code>	Draw li control serializ propert
≡ S	<code>ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT)</code>	Draw li control absolut positio
≡ S	<code>ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute)</code>	Draw li control absolut positio
≡ S	<code>ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListFlags)</code>	Draw li control absolut positio
≡ S	<code>ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, Single)</code>	Draw li control absolut positio
≡ S		

	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)	Draw li control absolut position
≡♦ S	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute, Single)	Draw li control absolut position
≡♦ S	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, Single, ReorderableListFlags)	Draw li control absolut position
≡♦ S	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute, Single, ReorderableListFlags)	Draw li control absolut position
≡♦ S	TextFieldItemDrawer	Draws allowin items to edited.
≡♦ S E	Title(String)	Draw ti control field.
≡♦ S E	Title(GUIContent)	Draw ti control field.
≡♦ S	Title(Rect, String)	Draw ti control field wi absolut position
≡♦ S		

[Title\(Rect, GUIContent\)](#)

Draw ti  
control  
field wi  
absolut  
positio

---

[Top](#)

## ► See Also

[Reference](#)

[ReorderableListGUI Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutCalculateListField Method

## ↳ Overload List

Name	Description
  <a href="#">CalculateListFieldHeight(Int32)</a>	Calculates height of field for absolute position.
  <a href="#">CalculateListFieldHeight(SerializedProperty)</a>	Calculates height of field for absolute position.
  <a href="#">CalculateListFieldHeight(IReorderableListAdapter)</a>	Calculates height of field for adapter collection.
  <a href="#">CalculateListFieldHeight(Int32, ReorderableListFlags)</a>	Calculates height of field for absolute position.
  <a href="#">CalculateListFieldHeight(Int32, Single)</a>	Calculates height of field for absolute position.

	position	
≡ S	CalculateListFieldHeight(SerializedProperty, ReorderableListFlags)	Calculate height of field for absolute position
≡ S	CalculateListFieldHeight(IReorderableListAdapter, ReorderableListFlags)	Calculate height of field for adapter collection
≡ S	CalculateListFieldHeight(Int32, Single, ReorderableListFlags)	Calculate height of field for absolute position

[Top](#)

## See Also

### Reference

[ReorderableListGUI Class](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutCalculateListField (Int32)

Calculate height of list field for absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static float CalculateListFieldHeight(  
    int itemCount  
)
```

### Parameters

*itemCount*

Type: **SystemInt32**

Count of items in list.

### Return Value

Type: **Single**

Required list height in pixels.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[CalculateListFieldHeight Overload](#)

[Rotorz.ReorderableList Namespace](#)



# ReorderableListGUILayoutCalculateListField Method (SerializedProperty)

Calculate height of list field for absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static float CalculateListFieldHeight(  
    SerializedProperty arrayProperty  
)
```

### Parameters

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

### Return Value

Type: **Single**

Required list height in pixels.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[CalculateListFieldHeight Overload](#)

[Rotorz.ReorderableList Namespace](#)



# ReorderableListGUILayoutCalculateListField Method (IReorderableListAdapter)

Calculate height of list field for adapted collection.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static float CalculateListFieldHeight(  
    IReorderableListAdapter adaptor  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

### Return Value

Type: [Single](#)

Required list height in pixels.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[CalculateListFieldHeight Overload](#)

[Rotorz.ReorderableList Namespace](#)



# ReorderableListGUILayoutCalculateListFieldHeight Method (Int32, ReorderableListFlags)

Calculate height of list field for absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static float CalculateListFieldHeight(  
    int itemCount,  
    ReorderableListFlags flags  
)
```

### Parameters

*itemCount*

Type: **SystemInt32**

Count of items in list.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

### Return Value

Type: **Single**

Required list height in pixels.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[CalculateListFieldHeight Overload](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutCalculateListField Method (Int32, Single)

Calculate height of list field for absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static float CalculateListFieldHeight(  
    int itemCount,  
    float itemHeight  
)
```

### Parameters

*itemCount*

Type: **SystemInt32**

Count of items in list.

*itemHeight*

Type: **SystemSingle**

Fixed height of list item.

### Return Value

Type: **Single**

Required list height in pixels.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[CalculateListFieldHeight Overload](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutCalculateListField Method (SerializedProperty, ReorderableListFlags)

Calculate height of list field for absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static float CalculateListFieldHeight(  
    SerializedProperty arrayProperty,  
    ReorderableListFlags flags  
)
```

## Parameters

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## Return Value

Type: **Single**

Required list height in pixels.

## ► See Also

## Reference

[ReorderableListGUI Class](#)

[CalculateListFieldHeight Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutCalculateListField Method (IReorderableListAdapter, ReorderableListFlags)

Calculate height of list field for adapted collection.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static float CalculateListFieldHeight(  
    IReorderableListAdapter adaptor,  
    ReorderableListFlags flags  
)
```

## Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)  
Reorderable list adaptor.

*flags*

Type: [Rotorz.ReorderableList.ReorderableListFlags](#)  
Optional flags to pass into list field.

## Return Value

Type: **Single**

Required list height in pixels.

## ► See Also

## Reference

[ReorderableListGUI Class](#)

[CalculateListFieldHeight Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutCalculateListFieldHeight Method (Int32, Single, ReorderableListFlags)

Calculate height of list field for absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static float CalculateListFieldHeight(  
    int itemCount,  
    float itemHeight,  
    ReorderableListFlags flags  
)
```

## Parameters

*itemCount*

Type: **SystemInt32**

Count of items in list.

*itemHeight*

Type: **SystemSingle**

Fixed height of list item.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## Return Value

Type: **Single**

Required list height in pixels.

## See Also

Reference

[ReorderableListGUI Class](#)

[CalculateListFieldHeight Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUIDefaultItemDrawer Method

Default list item drawer implementation.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#    UnityScript

[Copy](#)

```
public static T DefaultItemDrawer<T>(  
    Rect position,  
    T item  
)
```

## Parameters

*position*

Type: **Rect**

Position to draw list item control(s).

*item*

Type: **T**

Value of list item.

## Type Parameters

**T**

Type of list item.

## Return Value

Type: **T**

Unmodified value of list item.

## ◀ Remarks

Always presents the label "Item drawer not implemented.".

## ◀ See Also

[Reference](#)

[ReorderableListGUI Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout

## Method

### ▪ Overload List

	Name	Description
≡ S	<a href="#">ListField(SerializedProperty)</a>	Draw list field control for serializable property array.
≡ S	<a href="#">ListField(IReorderableListAdaptor)</a>	Draw list field control for adapted collection.
≡ S	<a href="#">ListFieldT(IListT, ReorderableListControlItemDrawerT)</a>	Draw list field control.
≡ S	<a href="#">ListField(SerializedProperty, ReorderableListControlDrawEmpty)</a>	Draw list field control for serializable property array.
≡ S	<a href="#">ListField(SerializedProperty, ReorderableListFlags)</a>	Draw list field control for serializable property array.
≡ S	<a href="#">ListField(SerializedProperty, Single)</a>	Draw list field control for serializable property array.

≡ S	ListField(IReorderableListAdaptor, ReorderableListControlDrawEmpty)	Draw list field control for adapted collection.
≡ S	ListField(IReorderableListAdaptor, ReorderableListFlags)	Draw list field control for adapted collection.
≡ S	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmpty)	Draw list field control.
≡ S	ListFieldT(IListT, ReorderableListControlItemDrawerT, ReorderableListFlags)	Draw list field control.
≡ S	ListFieldT(IListT, ReorderableListControlItemDrawerT, Single)	Draw list field control.
≡ S	ListField(SerializedProperty, ReorderableListControlDrawEmpty, ReorderableListFlags)	Draw list field control for serializable property array.
≡ S	ListField(SerializedProperty, Single, ReorderableListControlDrawEmpty)	Draw list field control for serializable property array.
≡ S	ListField(SerializedProperty, Single, ReorderableListFlags)	Draw list field control for serializable property array.
≡ S	ListField(IReorderableListAdaptor,	Draw list field

	<code>ReorderableListControlDrawEmpty,</code> <code>ReorderableListFlags)</code>	control for adapted collection.
 	<code>ListFieldT(IListT,</code> <code>ReorderableListControlItemDrawerT,</code> <code>ReorderableListControlDrawEmpty,</code> <code>ReorderableListFlags)</code>	Draw list field control.
 	<code>ListFieldT(IListT,</code> <code>ReorderableListControlItemDrawerT,</code> <code>ReorderableListControlDrawEmpty,</code> <code>Single)</code>	Draw list field control.
 	<code>ListFieldT(IListT,</code> <code>ReorderableListControlItemDrawerT,</code> <code>Single, ReorderableListFlags)</code>	Draw list field control.
 	<code>ListField(SerializedProperty, Single,</code> <code>ReorderableListControlDrawEmpty,</code> <code>ReorderableListFlags)</code>	Draw list field control for serializable property array.
 	<code>ListFieldT(IListT,</code> <code>ReorderableListControlItemDrawerT,</code> <code>ReorderableListControlDrawEmpty,</code> <code>Single, ReorderableListFlags)</code>	Draw list field control.

[Top](#)

## ▲ See Also

### Reference

[ReorderableListGUI Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayout Method (SerializedProperty)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(  
    SerializedProperty arrayProperty  
)
```

### Parameters

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

## ► See Also

Reference

[ReorderableListGUILayout Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayout Method (IReorderableListAdapter)

Draw list field control for adapted collection.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(  
    IReorderableListAdapter adaptor  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

## ► See Also

Reference

[ReorderableListGUILayout Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayout<T> Method (IList<T>, ReorderableListControlItemDrawer<T>)

Draw list field control.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField<T>(  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem  
)
```

## Parameters

*list*

Type: **System.Collections.Generic.IList<T>**

The list which can be reordered.

*drawItem*

Type: **Rotorz.ReorderableList.ReorderableListControlItemDrawer<T>**

Callback to draw list item.

## Type Parameters

*T*

Type of list item.

## ◀ See Also

### Reference

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout Method (SerializedProperty, ReorderableListControlDrawEmpty)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(  
    SerializedProperty arrayProperty,  
    ReorderableListControlDrawEmpty drawEmpty  
)
```

## Parameters

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmpty](#)

Callback to draw custom content for empty list (optional).

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[ListField Overload](#)

## Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout Method (SerializedProperty, ReorderableListFlags)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(  
    SerializedProperty arrayProperty,  
    ReorderableListFlags flags  
)
```

### Parameters

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[ListField Overload](#)

## Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout Method (SerializedProperty, Single)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(  
    SerializedProperty arrayProperty,  
    float fixedItemHeight  
)
```

### Parameters

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*fixedItemHeight*

Type: **SystemSingle**

Use fixed height for items rather than

**GetPropertyHeight(SerializedProperty)**.

## ► See Also

Reference

[ReorderableListGUILayout Class](#)

ListField Overload  
Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutMethod (IReorderableListAdapter, ReorderableListControlDrawEmpty)

Draw list field control for adapted collection.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(
    IReorderableListAdapter adaptor,
    ReorderableListControlDrawEmpty drawEmpty
)
```

## Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

*drawEmpty*

Type: [Rotorz.ReorderableList.ReorderableListControlDrawEmpty](#)

Callback to draw custom content for empty list (optional).

## ► See Also

Reference

[ReorderableListGUILayout Class](#)

[ListField Overload](#)

## Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout Method (IReorderableListAdapter, ReorderableListFlags)

Draw list field control for adapted collection.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(  
    IReorderableListAdapter adaptor,  
    ReorderableListFlags flags  
)
```

### Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)  
Reorderable list adaptor.

*flags*

Type: [Rotorz.ReorderableList.ReorderableListFlags](#)  
Optional flags to pass into list field.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[ListField Overload](#)

## Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout<T> Method (IList<T>, ReorderableListControlItemDrawer<T> ReorderableListControlDrawEmpty)

Draw list field control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static void ListField<T>(  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem,  
    ReorderableListControlDrawEmpty drawEmpty  
)
```

## Parameters

*list*

Type: [System.Collections.Generic.IList<T>](#)

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer<T>](#)

Callback to draw list item.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmpty](#)

Callback to draw custom content for empty list (optional).

## Type Parameters

*T*

Type of list item.

## ► See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout<T> Method (IList<T>, ReorderableListControlItemDrawer<T> ReorderableListFlags)

Draw list field control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField<T>(  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem,  
    ReorderableListFlags flags  
)
```

## Parameters

*list*

Type: [System.Collections.Generic.IList<T>](#)

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer<T>](#)

Callback to draw list item.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## Type Parameters

*T*

Type of list item.

## ► See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout<T> Method (IList<T>, ReorderableListControlItemDrawer<T> Single)

Draw list field control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField<T>(  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem  
    float itemHeight  
)
```

## Parameters

*list*

Type: [System.Collections.Generic.IList<T>](#)

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer<T>](#)

Callback to draw list item.

*itemHeight*

Type: [System.Single](#)

Height of a single list item.

## Type Parameters

*T*

Type of list item.

## ► See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout Method (SerializedProperty, ReorderableListControlDrawEmpty, ReorderableListFlags)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(  
    SerializedProperty arrayProperty,  
    ReorderableListControlDrawEmpty drawEmpty,  
    ReorderableListFlags flags  
)  

```

## Parameters

*arrayProperty*

Type: [SerializedProperty](#)

Serializable property.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmpty](#)

Callback to draw custom content for empty list (optional).

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## ◀ See Also

### Reference

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutMethod (SerializedProperty, Single, ReorderableListControlDrawEmpty)

Draw list field control for serializable property array.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(  
    SerializedProperty arrayProperty,  
    float fixedItemHeight,  
    ReorderableListControlDrawEmpty drawEmpty  
)  
{ }  
[ ]
```

## Parameters

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*fixedItemHeight*

Type: **SystemSingle**

Use fixed height for items rather than

**GetPropertyHeight(SerializedProperty)**.

*drawEmpty*

Type: **Rotorz.ReorderableListReorderableListControlDrawEmpty**

Callback to draw custom content for empty list (optional).

## ◀ See Also

### Reference

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout Method (SerializedProperty, Single, ReorderableListFlags)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(  
    SerializedProperty arrayProperty,  
    float fixedItemHeight,  
    ReorderableListFlags flags  
)
```

## Parameters

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*fixedItemHeight*

Type: **SystemSingle**

Use fixed height for items rather than

**GetPropertyHeight(SerializedProperty).**

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## ◀ See Also

### Reference

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutMethod (IReorderableListAdapter, ReorderableListControlDrawEmpty, ReorderableListFlags)

Draw list field control for adapted collection.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(  
    IReorderableListAdapter adaptor,  
    ReorderableListControlDrawEmpty drawEmpty,  
    ReorderableListFlags flags  
)  
{  
    [1]                          [2]  
}
```

## Parameters

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

*drawEmpty*

Type: [Rotorz.ReorderableList.ReorderableListControlDrawEmpty](#)

Callback to draw custom content for empty list (optional).

*flags*

Type: [Rotorz.ReorderableList.ReorderableListFlags](#)

Optional flags to pass into list field.

## ◀ See Also

### Reference

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout<T> Method (IList<T>, ReorderableListControlItemDrawer<T> ReorderableListControlDrawEmpty, ReorderableListFlags)

Draw list field control.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static void ListField<T>(  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem,  
    ReorderableListControlDrawEmpty drawEmpty,  
    ReorderableListFlags flags  
)
```

## Parameters

*list*

Type: [System.Collections.Generic.IList<T>](#)

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer<T>](#)

Callback to draw list item.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmpty](#)

Callback to draw custom content for empty list (optional).

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## Type Parameters

*T*

Type of list item.

## ▲ See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout<T> Method (IList<T>, ReorderableListControlItemDrawer<T> ReorderableListControlDrawEmpty, Single)

Draw list field control.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static void ListField<T>(  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem,  
    ReorderableListControlDrawEmpty drawEmpty,  
    float itemHeight  
)
```

## Parameters

*list*

Type: [System.Collections.Generic.IList<T>](#)

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer<T>](#)

Callback to draw list item.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmpty](#)

Callback to draw custom content for empty list (optional).

*itemHeight*

Type: **SystemSingle**

Height of a single list item.

## Type Parameters

*T*

Type of list item.

## ▲ See Also

Reference

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout<T> Method (IList<T>, ReorderableListControlItemDrawer<T> Single, ReorderableListFlags)

Draw list field control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField<T>(  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem,  
    float itemHeight,  
    ReorderableListFlags flags  
)
```

## Parameters

*list*

Type: [System.Collections.Generic.IList<T>](#)

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer<T>](#)

Callback to draw list item.

*itemHeight*

Type: [System.Single](#)

Height of a single list item.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## Type Parameters

*T*

Type of list item.

## ◀ See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutMethod (SerializedProperty, Single, ReorderableListControlDrawEmpty, ReorderableListFlags)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListField(  
    SerializedProperty arrayProperty,  
    float fixedItemHeight,  
    ReorderableListControlDrawEmpty drawEmpty,  
    ReorderableListFlags flags  
)
```

## Parameters

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*fixedItemHeight*

Type: **SystemSingle**

Use fixed height for items rather than

**GetPropertyHeight(SerializedProperty).**

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmpty](#)  
Callback to draw custom content for empty list (optional).

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)  
Optional flags to pass into list field.

## See Also

### Reference

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout<T> Method (IList<T>, ReorderableListControlItemDrawer<T> ReorderableListControlDrawEmpty, Single, ReorderableListFlags)

Draw list field control.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static void ListField<T>(  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem,  
    ReorderableListControlDrawEmpty drawEmpty,  
    float itemHeight,  
    ReorderableListFlags flags  
)
```

## Parameters

*list*

Type: **System.Collections.Generic.IList<T>**

The list which can be reordered.

*drawItem*

Type: **Rotorz.ReorderableListReorderableListControlItemDrawer<T>**

Callback to draw list item.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmpty](#)

Callback to draw custom content for empty list (optional).

*itemHeight*

Type: **SystemSingle**

Height of a single list item.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## Type Parameters

*T*

Type of list item.

## ▲ See Also

### Reference

[ReorderableListGUI Class](#)

[ListField Overload](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayoutAbsolute Method

## ↳ Overload List

Name	Description
  <a href="#">ListFieldAbsolute(Rect, IReorderableListAdapter)</a>	Draw list field control for adapted collection.
  <a href="#">ListFieldAbsolute(Rect, SerializedProperty)</a>	Draw list field control for serializable property array.
  <a href="#">ListFieldAbsolute(Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAbsolute)</a>	Draw list field control for adapted collection.
  <a href="#">ListFieldAbsolute(Rect, IReorderableListAdapter, ReorderableListFlags)</a>	Draw list field control for adapted collection.
  <a href="#">ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT)</a>	Draw list field control with absolute positioning.

≡ S	ListFieldAbsolute(Rect, SerializedProperty, ReorderableListControlDrawEmptyAbsolute)	Draw list field control for serializable property array.
≡ S	ListFieldAbsolute(Rect, SerializedProperty, ReorderableListFlags)	Draw list field control for serializable property array.
≡ S	ListFieldAbsolute(Rect, SerializedProperty, Single)	Draw list field control for serializable property array.
≡ S	ListFieldAbsolute(Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)	Draw list field control for adapted collection.
≡ S	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute)	Draw list field control with absolute positioning.
≡ S	ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListFlags)	Draw list field control with absolute positioning.
≡ S	ListFieldAbsoluteT(Rect, IListT,	Draw list

	<code>ReorderableListControlItemDrawerT, Single)</code>	field control with absolute positioning.
≡ S	<code>ListFieldAbsolute(Rect, SerializedProperty, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)</code>	Draw list field control for serializable property array.
≡ S	<code>ListFieldAbsolute(Rect, SerializedProperty, Single, ReorderableListControlDrawEmptyAbsolute)</code>	Draw list field control for serializable property array.
≡ S	<code>ListFieldAbsolute(Rect, SerializedProperty, Single, ReorderableListFlags)</code>	Draw list field control for serializable property array.
≡ S	<code>ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)</code>	Draw list field control with absolute positioning.
≡ S	<code>ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute, Single)</code>	Draw list field control with absolute positioning.
≡ S	<code>ListFieldAbsoluteT(Rect, IListT,</code>	Draw list

`ReorderableListControlItemDrawerT, Single, ReorderableListFlags)`

field control with absolute positioning.



`ListFieldAbsolute(Rect, SerializedProperty, Single, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)`

Draw list field control for serializable property array.



`ListFieldAbsoluteT(Rect, IListT, ReorderableListControlItemDrawerT, ReorderableListControlDrawEmptyAbsolute, Single, ReorderableListFlags)`

Draw list field control with absolute positioning.

[Top](#)

## ▲ See Also

[Reference](#)

[ReorderableListGUI Class](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, IReorderableListAdapter)

Draw list field control for adapted collection.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute(  
    Rect position,  
    IReorderableListAdapter adaptor  
)
```

### Parameters

*position*

Type: [Rect](#)

Position of control.

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[ListFieldAbsolute Overload](#)

## Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, SerializedProperty)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute(  
    Rect position,  
    SerializedProperty arrayProperty  
)
```

### Parameters

*position*

Type: **Rect**

Position of control.

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAction)

Draw list field control for adapted collection.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute(  
    Rect position,  
    IReorderableListAdapter adaptor,  
    ReorderableListControlDrawEmptyAbsolute drawEmpty)  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

*drawEmpty*

Type: [Rotorz.ReorderableList.ReorderableListControlDrawEmptyAction](#)

Callback to draw custom content for empty list (optional).

## ◀ See Also

### Reference

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, IReorderableListAdapter, ReorderableListFlags)

Draw list field control for adapted collection.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#

UnityScript

Copy

```
public static void ListFieldAbsolute(  
    Rect position,  
    IReorderableListAdapter adaptor,  
    ReorderableListFlags flags  
)
```

## Parameters

*position*

Type: [Rect](#)

Position of control.

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

*flags*

Type: [Rotorz.ReorderableList.ReorderableListFlags](#)

Optional flags to pass into list field.

## ◀ See Also

### Reference

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, IList<T>, ReorderableListControlItemDrawer<T>)

Draw list field control with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute<T>(  
    Rect position,  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*list*

Type: **System.Collections.GenericIList<T>**

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer<T>](#)

Callback to draw list item.

## Type Parameters

*T*

Type of list item.

## ► See Also

Reference

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, SerializedProperty, ReorderableListControlDrawEmptyA)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute(  
    Rect position,  
    SerializedProperty arrayProperty,  
    ReorderableListControlDrawEmptyAbsolute drawEmpty  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmptyA](#)

Callback to draw custom content for empty list (optional).

## ◀ See Also

### Reference

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, SerializedProperty, ReorderableListFlags)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute(  
    Rect position,  
    SerializedProperty arrayProperty,  
    ReorderableListFlags flags  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## ► See Also

## Reference

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, SerializedProperty, Single)

Draw list field control for serializable property array.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute(  
    Rect position,  
    SerializedProperty arrayProperty,  
    float fixedItemHeight  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*fixedItemHeight*

Type: **SystemSingle**

Use fixed height for items rather than

**GetPropertyHeight(SerializedProperty)**.

## ◀ See Also

### Reference

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsoluteMethod (Rect, IReorderableListAdapter, ReorderableListControlDrawEmptyAction, ReorderableListFlags)

Draw list field control for adapted collection.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute(  
    Rect position,  
    IReorderableListAdapter adaptor,  
    ReorderableListControlDrawEmptyAbsolute drawEmpty,  
    ReorderableListFlags flags  
)
```

## Parameters

*position*

Type: [Rect](#)

Position of control.

*adaptor*

Type: [Rotorz.ReorderableList.IReorderableListAdapter](#)

Reorderable list adaptor.

*drawEmpty*

Type: [Rotorz.ReorderableList.ReorderableListControlDrawEmptyAction](#)

Callback to draw custom content for empty list (optional).

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## ◀ See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, IList<T>, ReorderableListControlItemDrawer<T>, ReorderableListControlDrawEmptyAction)

Draw list field control with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute<T>(  
    Rect position,  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem,  
    ReorderableListControlDrawEmptyAbsolute action  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*list*

Type: **System.Collections.Generic.IList<T>**

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer<T>](#)

Callback to draw list item.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmptyA](#)

Callback to draw custom content for empty list (optional).

## Type Parameters

*T*

Type of list item.

## ◀ See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, IList<T>, ReorderableListControlItemDrawer<T>, ReorderableListFlags)

Draw list field control with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#

UnityScript

[Copy](#)

```
public static void ListFieldAbsolute<T>(  
    Rect position,  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem,  
    ReorderableListFlags flags  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*list*

Type: **System.Collections.GenericIList<T>**

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer<T>](#)

Callback to draw list item.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## Type Parameters

*T*

Type of list item.

## ◀ See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, IList<T>, ReorderableListControlItemDrawer<T> Single)

Draw list field control with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute<T>(  
    Rect position,  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem,  
    float itemHeight  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*list*

Type: **System.Collections.GenericIList<T>**

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer<T>](#)

Callback to draw list item.

*itemHeight*

Type: **SystemSingle**

Height of a single list item.

## Type Parameters

*T*

Type of list item.

## ◀ See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, SerializedProperty, ReorderableListControlDrawEmptyA, ReorderableListFlags)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute(  
    Rect position,  
    SerializedProperty arrayProperty,  
    ReorderableListControlDrawEmptyAbsolute drawEmpty,  
    ReorderableListFlags flags  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmptyA](#)

Callback to draw custom content for empty list (optional).

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## ◀ See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, SerializedProperty, SystemSingle)

## ReorderableListControlDrawEmptyArea

Draw list field control for serializable property array.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

### ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute(  
    Rect position,  
    SerializedProperty arrayProperty,  
    float fixedItemHeight,  
    ReorderableListControlDrawEmptyArea  
)
```

### Parameters

*position*

Type: **Rect**

Position of control.

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*fixedItemHeight*

Type: **SystemSingle**

Use fixed height for items rather than

**GetPropertyHeight(SerializedProperty)**.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmptyA](#)  
Callback to draw custom content for empty list (optional).

## See Also

Reference

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, SerializedProperty, Single, ReorderableListFlags)

Draw list field control for serializable property array.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute(  
    Rect position,  
    SerializedProperty arrayProperty,  
    float fixedItemHeight,  
    ReorderableListFlags flags  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*fixedItemHeight*

Type: **SystemSingle**

Use fixed height for items rather than

**GetPropertyHeight(SerializedProperty)**.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)  
Optional flags to pass into list field.

## See Also

### Reference

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, IList<T>, ReorderableListControlItemDrawer<T>, ReorderableListControlDrawEmptyAbsolute, ReorderableListFlags)

Draw list field control with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute<T>(
    Rect position,
    IList<T> list,
    ReorderableListControlItemDrawer<T> drawer,
    ReorderableListControlDrawEmptyAbsolute drawEmpty,
    ReorderableListFlags flags
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*list*

Type: **System.Collections.Generic.IList<T>**

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer](#)*T*

Callback to draw list item.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmpty](#)*A*

Callback to draw custom content for empty list (optional).

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## Type Parameters

*T*

Type of list item.

## ▲ See Also

### Reference

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayoutAbsolute Method (Rect, IList<T>, ReorderableListControlItemDrawer<T>, ReorderableListControlDrawEmptyActionSingle)

Draw list field control with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void ListFieldAbsolute<T>(
    Rect position,
    IList<T> list,
    ReorderableListControlItemDrawer<T> drawer,
    ReorderableListControlDrawEmptyAbsolute action,
    float itemHeight
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*list*

Type: **System.Collections.Generic.IList<T>**

The list which can be reordered.

#### *drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer{T}](#)

Callback to draw list item.

#### *drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmpty{A}](#)

Callback to draw custom content for empty list (optional).

#### *itemHeight*

Type: **SystemSingle**

Height of a single list item.

## Type Parameters

### *T*

Type of list item.

## ▲ See Also

### Reference

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayoutAbsolute Method (Rect, IList<T>, ReorderableListControlItemDrawer<T>, Single, ReorderableListFlags)

Draw list field control with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#

UnityScript

[Copy](#)

```
public static void ListFieldAbsolute<T>(  
    Rect position,  
    IList<T> list,  
    ReorderableListControlItemDrawer<T> drawItem,  
    float itemHeight,  
    ReorderableListFlags flags  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*list*

Type: **System.Collections.GenericIList<T>**

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer](#)  
*T*  
Callback to draw list item.

*itemHeight*

Type: **SystemSingle**  
Height of a single list item.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)  
Optional flags to pass into list field.

## Type Parameters

*T*

Type of list item.

## See Also

### Reference

[ReorderableListGUI Class](#)  
[ListFieldAbsolute Overload](#)  
[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayoutAbsolute Method (Rect, SerializedProperty, Si ReorderableListControlDrawEmptyA ReorderableListFlags)

Draw list field control for serializable property array.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#

UnityScript

[Copy](#)

```
public static void ListFieldAbsolute(  
    Rect position,  
    SerializedProperty arrayProperty,  
    float fixedItemHeight,  
    ReorderableListControlDrawEmptyAbsolute c  
    ReorderableListFlags flags  
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*arrayProperty*

Type: **SerializedProperty**

Serializable property.

*fixedItemHeight*

Type: **SystemSingle**

Use fixed height for items rather than  
**GetPropertyHeight(SerializedProperty)**.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmptyA](#)

Callback to draw custom content for empty list (optional).

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## ▲ See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutAbsolute Method (Rect, IList<T>, ReorderableListControlItemDrawer<T>, ReorderableListControlDrawEmptyAbsolute, Single, ReorderableListFlags)

Draw list field control with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#

UnityScript

[Copy](#)

```
public static void ListFieldAbsolute<T>(
    Rect position,
    IList<T> list,
    ReorderableListControlItemDrawer<T> drawer,
    ReorderableListControlDrawEmptyAbsolute drawEmpty,
    float itemHeight,
    ReorderableListFlags flags
)
```

## Parameters

*position*

Type: **Rect**

Position of control.

*list*

Type: **System.Collections.GenericIList***T*

The list which can be reordered.

*drawItem*

Type: [Rotorz.ReorderableListReorderableListControlItemDrawer](#)*T*

Callback to draw list item.

*drawEmpty*

Type: [Rotorz.ReorderableListReorderableListControlDrawEmpty](#)*A*

Callback to draw custom content for empty list (optional).

*itemHeight*

Type: **SystemSingle**

Height of a single list item.

*flags*

Type: [Rotorz.ReorderableListReorderableListFlags](#)

Optional flags to pass into list field.

## Type Parameters

*T*

Type of list item.

## See Also

[Reference](#)

[ReorderableListGUI Class](#)

[ListFieldAbsolute Overload](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUITextFieldItemDrawer Method

Draws text field allowing list items to be edited.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static string TextFieldItemDrawer(  
    Rect position,  
    string item  
)
```

### Parameters

*position*

Type: **Rect**

Position to draw list item control(s).

*item*

Type: **SystemString**

Value of list item.

### Return Value

Type: **String**

Modified value of list item.

## ► Remarks

Null values are automatically changed to empty strings since null values cannot be edited using a text field.

Value of `GUI.changed` is set to `true` if value of item is modified.

## ◀ See Also

[Reference](#)

[ReorderableListGUI Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout Method

## ▪ Overload List

	Name	Description
  	<a href="#">Title(String)</a>	Draw title control for list field.
  	<a href="#">Title(GUIContent)</a>	Draw title control for list field.
 	<a href="#">Title(Rect, String)</a>	Draw title control for list field with absolute positioning.
 	<a href="#">Title(Rect, GUIContent)</a>	Draw title control for list field with absolute positioning.

[Top](#)

## ▪ See Also

### Reference

[ReorderableListGUILayout Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayout Method (String)

Draw title control for list field.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static void Title(  
    string title  
)
```

### Parameters

*title*

Type: **System.String**

Text for title control.

## ► Remarks

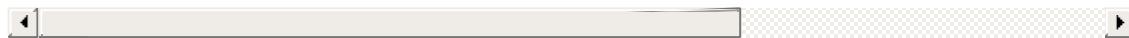
When needed, should be shown immediately before list field.

## ► Examples

C#    UnityScript

Copy

```
ReorderableListGUI.Title("Your Title");  
ReorderableListGUI.ListField(list, DynamicListGU.
```



## ◀ See Also

Reference

[ReorderableListGUI Class](#)

[Title Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout Method (GUIContent)

Draw title control for list field.

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static void Title(  
    GUIContent title  
)
```

### Parameters

*title*

Type: **GUIContent**

Content for title control.

## ► Remarks

When needed, should be shown immediately before list field.

## ► Examples

C#    UnityScript

Copy

```
ReorderableListGUI.Title(titleContent);  
ReorderableListGUI.ListField(list, DynamicListGU.
```



## ◀ See Also

Reference

[ReorderableListGUI Class](#)

[Title Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout Method (Rect, String)

Draw title control for list field with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void Title(  
    Rect position,  
    string text  
)
```

### Parameters

*position*

Type: **Rect**

Position of control.

*text*

Type: **SystemString**

Text for title control.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[Title Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayout Method (Rect, GUIContent)

Draw title control for list field with absolute positioning.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static void Title(  
    Rect position,  
    GUIContent title  
)
```

### Parameters

*position*

Type: **Rect**

Position of control.

*title*

Type: **GUIContent**

Content for title control.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[Title Overload](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUI Properties

The [ReorderableListGUI](#) type exposes the following members.

## Properties

	Name	Description
 	<a href="#">CurrentItemIndex</a>	Gets the zero-based index of the list item that is currently being drawn; or a value of -1 if no item is currently being drawn.
 	<a href="#">CurrentItemTotalPosition</a>	Gets the total position of the list item that is currently being drawn.
 	<a href="#">CurrentListControlID</a>	Gets the control ID of the list that is currently being drawn.
 	<a href="#">CurrentListPosition</a>	Gets the position of the list control that is currently being drawn.
 	<a href="#">IndexOfChangedItem</a>	Gets or sets the zero-based index of the last item that was changed. A value of -1 indicates that no item was changed by list.

[Top](#)

## ◀ See Also

### Reference

[ReorderableListGUI Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListGUILayoutCurrentItemIndex Property

Gets the zero-based index of the list item that is currently being drawn; or a value of -1 if no item is currently being drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static int CurrentItemIndex { get; }
```

Property Value

Type: **Int32**

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayoutCurrentItemTotalPosition Property

Gets the total position of the list item that is currently being drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static Rect CurrentItemTotalPosition { get
```



Property Value

Type: [Rect](#)

## ► Remarks

The value of this property should be ignored for [Layout](#) type events when using reorderable list controls with automatic layout.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayoutCurrentListConti Property

Gets the control ID of the list that is currently being drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#    UnityScript

Copy

```
public static int CurrentListControlID { get; }
```

Property Value

Type: [Int32](#)

## ◀ See Also

Reference

[ReorderableListGUILayout Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayoutCurrentListPosit Property

Gets the position of the list control that is currently being drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static Rect CurrentListPosition { get; }
```

Property Value

Type: **Rect**

## ► Remarks

The value of this property should be ignored for **Layout** type events when using reorderable list controls with automatic layout.

## ► See Also

Reference

[ReorderableListGUILayout Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListGUILayoutIndexOfChangedItem Property

Gets or sets the zero-based index of the last item that was changed. A value of -1 indicates that no item was changed by list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

Copy

```
public static int IndexOfChangedItem { get; }
```

Property Value

Type: [Int32](#)

## ► Remarks

This property should not be set when items are added or removed.

## ► See Also

[Reference](#)

[ReorderableListGUILayout Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListStyles Class

Styles for the [ReorderableListControl](#).

## ► Inheritance Hierarchy

**SystemObject** [Rotorz.ReorderableListReorderableListStyles](#)

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static class ReorderableListStyles
```

The [ReorderableListStyles](#) type exposes the following members.

## ► Properties

	Name	Description
 	<a href="#">Container</a>	Gets style for the background of list control.
 	<a href="#">Container2</a>	Gets an alternative style for the background of list control.
 	<a href="#">FooterButton</a>	Gets style for footer button.
 	<a href="#">FooterButton2</a>	Gets an alternative style

for footer button.

 	<a href="#">HorizontalLineColor</a>	Gets color for the horizontal lines that appear between list items.
 	<a href="#">ButtonItem</a>	Gets style for remove item button.
 	<a href="#">SelectedItem</a>	Gets style for the background of a selected item.
 	<a href="#">SelectionBackgroundColor</a>	Gets color of background for a selected list item.
 	<a href="#">Title</a>	Gets style for title header.

[Top](#)

## ▲ See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListStyles Properties

The [ReorderableListStyles](#) type exposes the following members.

## Properties

	Name	Description
 	<a href="#">Container</a>	Gets style for the background of list control.
 	<a href="#">Container2</a>	Gets an alternative style for the background of list control.
 	<a href="#">FooterButton</a>	Gets style for footer button.
 	<a href="#">FooterButton2</a>	Gets an alternative style for footer button.
 	<a href="#">HorizontalLineColor</a>	Gets color for the horizontal lines that appear between list items.
 	<a href="#">ButtonItem</a>	Gets style for remove item button.
 	<a href="#">SelectedItem</a>	Gets style for the background of a selected item.
 	<a href="#">SelectionBackgroundColor</a>	Gets color of background for a selected list item.



## Title

Gets style for title header.

[Top](#)

## ▲ See Also

### Reference

[ReorderableListStyles Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# ReorderableListStylesContainer Property

Gets style for the background of list control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    [UnityScript](#)

[Copy](#)

```
public static GUIStyle Container { get; }
```

Property Value

Type: **GUIStyle**

## ► See Also

Reference

[ReorderableListStyles Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListStylesContainer2 Property

Gets an alternative style for the background of list control.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#    UnityScript

Copy

```
public static GUIStyle Container2 { get; }
```

Property Value

Type: **GUIStyle**

## ◀ See Also

Reference

[ReorderableListStyles Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListStylesFooterButton Property

Gets style for footer button.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    [UnityScript](#)

[Copy](#)

```
public static GUIStyle FooterButton { get; }
```

Property Value

Type: **GUIStyle**

## ► See Also

Reference

[ReorderableListStyles Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListStylesFooterButton2 Property

Gets an alternative style for footer button.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#    UnityScript

Copy

```
public static GUIStyle FooterButton2 { get; }
```

Property Value

Type: **GUIStyle**

## ◀ See Also

Reference

[ReorderableListStyles Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListStylesHorizontalLine Property

Gets color for the horizontal lines that appear between list items.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static Color HorizontalLineColor { get; }
```

Property Value

Type: **Color**

## ► See Also

Reference

[ReorderableListStyles Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListStylesButtonItemButton Property

Gets style for remove item button.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static GUIStyle ItemButton { get; }
```

Property Value

Type: **GUIStyle**

## ► See Also

Reference

[ReorderableListStyles Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListStylesSelectedItem Property

Gets style for the background of a selected item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ◀ Syntax

C#    UnityScript

[Copy](#)

```
public static GUIStyle SelectedItem { get; }
```

Property Value

Type: **GUIStyle**

## ◀ See Also

Reference

[ReorderableListStyles Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListStylesSelectionBack Property

Gets color of background for a selected list item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static Color SelectionBackgroundColor { get
```

Property Value

Type: [Color](#)

## ► See Also

[Reference](#)

[ReorderableListStyles Class](#)

[Rotorz.ReorderableList Namespace](#)

# ReorderableListStylesTitle Property

Gets style for title header.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public static GUIStyle Title { get; }
```

Property Value

Type: **GUIStyle**

## ► See Also

Reference

[ReorderableListStyles Class](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptor Class

Reorderable list adaptor for serialized array property.

## ► Inheritance Hierarchy

**SystemObject** Rotorz.ReorderableListSerializedPropertyAdaptor

**Namespace:** Rotorz.ReorderableList

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public class SerializedPropertyAdaptor : IReorderer
```

The SerializedPropertyAdaptor type exposes the following members.

## ► Constructors

Name	Description
 <a href="#">SerializedPropertyAdaptor(SerializedProperty)</a>	Initializes a new instance of the SerializedPropertyAdaptor class.
 <a href="#">SerializedPropertyAdaptor(SerializedProperty, Single)</a>	Initializes a new instance of the SerializedPropertyAdaptor class.

[Top](#)

## ▪ Methods

	Name	Description
≡■	Add	Add new element at end of list.
≡■	BeginGUI	Occurs before any list items are drawn.
≡■	CanDrag	Determines whether an item can be reordered by dragging mouse.
≡■	CanRemove	Determines whether an item can be removed from list.
≡■	Clear	Clear all elements from list.
≡■	DrawItem	Draws main interface for a list item.
≡■	DrawItemBackground	Draws background of a list item.
≡■	Duplicate	Duplicate existing element.
≡■	EndGUI	Occurs after all list items have been drawn.
≡■	GetItemHeight	Gets height of list item in pixels.
≡■	Insert	Insert new element at specified index.
≡■	Move	Move element from source index to destination index.



## Remove

Remove element at specified index.

[Top](#)

## Fields

	Name	Description
◆	<a href="#">FixedItemHeight</a>	Fixed height of each list item.

[Top](#)

## Properties

	Name	Description
	<a href="#">arrayProperty</a>	Gets the underlying serialized array property.
	<a href="#">Count</a>	Gets count of elements in list.
	<a href="#">Item</a>	Gets element from list.

[Top](#)

## Remarks

This adaptor can be subclassed to add special logic to item height calculation. You may want to implement a custom adaptor class where specialised functionality is needed.

List elements are **not** cloned using the **ICloneable** interface when using a **SerializedProperty** to manipulate lists.

## See Also

[Reference](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# SerializedPropertyAdaptor Constructor

## ► Overload List

Name	Description
 <a href="#">SerializedPropertyAdaptor(SerializedProperty)</a>	Initializes a new instance of the SerializedPropertyAdaptor class.
 <a href="#">SerializedPropertyAdaptor(SerializedProperty, Single)</a>	Initializes a new instance of the SerializedPropertyAdaptor class.

[Top](#)

## ► See Also

### Reference

[SerializedPropertyAdaptor Class](#)  
[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptor Constructor (SerializedProperty)

Initializes a new instance of [SerializedPropertyAdaptor](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public SerializedPropertyAdaptor(  
    SerializedProperty arrayProperty  
)
```

### Parameters

*arrayProperty*

Type: **SerializedProperty**

Serialized property for entire array.

## ► See Also

### Reference

[SerializedPropertyAdaptor Class](#)

[SerializedPropertyAdaptor Overload](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptor

## Constructor (SerializedProperty, Single)

Initializes a new instance of [SerializedPropertyAdaptor](#).

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

### ► Syntax

C#    UnityScript

[Copy](#)

```
public SerializedPropertyAdaptor(  
    SerializedProperty arrayProperty,  
    float fixedItemHeight  
)
```

### Parameters

*arrayProperty*

Type: **SerializedProperty**

Serialized property for entire array.

*fixedItemHeight*

Type: **SystemSingle**

Non-zero height overrides property drawer height calculation.

### ► See Also

[Reference](#)

[SerializedPropertyAdaptor Class](#)

[SerializedPropertyAdaptor Overload](#)

## Rotorz.ReorderableList Namespace

---

©2013-2015 Rotorz Limited. All rights reserved.

# SerializedPropertyAdaptor Fields

The [SerializedPropertyAdaptor](#) type exposes the following members.

## Fields

	Name	Description
◆	<a href="#">FixedItemHeight</a>	Fixed height of each list item.

[Top](#)

## See Also

[Reference](#)

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# SerializedPropertyAdaptorFixedItemField

Fixed height of each list item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public float FixedItemHeight
```

Field Value

Type: **Single**

## ► Remarks

Non-zero value overrides property drawer height calculation which is more efficient.

## ► See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptor Methods

The [SerializedPropertyAdaptor](#) type exposes the following members.

## ▪ Methods

	Name	Description
≡	<a href="#">Add</a>	Add new element at end of list.
≡	<a href="#">BeginGUI</a>	Occurs before any list items are drawn.
≡	<a href="#">CanDrag</a>	Determines whether an item can be reordered by dragging mouse.
≡	<a href="#">CanRemove</a>	Determines whether an item can be removed from list.
≡	<a href="#">Clear</a>	Clear all elements from list.
≡	<a href="#">DrawItem</a>	Draws main interface for a list item.
≡	<a href="#">DrawItemBackground</a>	Draws background of a list item.
≡	<a href="#">Duplicate</a>	Duplicate existing element.
≡	<a href="#">EndGUI</a>	Occurs after all list items have been drawn.

≡	<a href="#">GetItemHeight</a>	Gets height of list item in pixels.
≡	<a href="#">Insert</a>	Insert new element at specified index.
≡	<a href="#">Move</a>	Move element from source index to destination index.
≡	<a href="#">Remove</a>	Remove element at specified index.

---

[Top](#)

## ▲ See Also

### Reference

[SerializedPropertyAdaptor Class](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# SerializedPropertyAdaptorAdd Method

Add new element at end of list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    [UnityScript](#)

[Copy](#)

```
public void Add()
```

Implements

[IReorderableListAdaptorAdd](#)

## ► See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptorBeginGUI Method

Occurs before any list items are drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void BeginGUI()
```

Implements

[IReorderableListAdaptorBeginGUI](#)

## ► Remarks

This method is only used to handle GUI repaint events.

## ► See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptorCanDrag Method

Determines whether an item can be reordered by dragging mouse.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual bool CanDrag(  
    int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index for list element.

### Return Value

Type: **Boolean**

A value of **true** if item can be dragged; otherwise **false**.

### Implements

[IReorderableListAdaptorCanDrag\(Int32\)](#)

## ► Remarks

This should be a light-weight method since it will be used to determine whether grab handle should be included for each item in a reorderable list.

Please note that returning a value of `false` does not prevent movement on list item since other draggable items can be moved around it.

## See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# SerializedPropertyAdaptorCanRemove Method

Determines whether an item can be removed from list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual bool CanRemove(  
    int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index for list element.

### Return Value

Type: **Boolean**

A value of **true** if item can be removed; otherwise **false**.

### Implements

[IReorderableListAdaptorCanRemove\(Int32\)](#)

## ► Remarks

This should be a light-weight method since it will be used to determine whether remove button should be included for each item in list.

This is redundant when [HideRemoveButtons](#) is specified.

## ◀ See Also

[Reference](#)

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# SerializedPropertyAdaptorClear Method

Clear all elements from list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public void Clear()
```

Implements

[IReorderableListAdaptorClear](#)

## ► See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptorDrawItem Method

Draws main interface for a list item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void DrawItem(  
    Rect position,  
    int index  
)
```

### Parameters

*position*

Type: **Rect**

Position in GUI.

*index*

Type: **SystemInt32**

Zero-based index of array element.

### Implements

[IReorderableListAdaptorDrawItem\(Rect, Int32\)](#)

## ► Remarks

This method is used to handle all GUI events.

## ◀ See Also

Reference

[SerializedPropertyAdaptor Class](#)  
[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# SerializedPropertyAdaptorDrawItem Method

Draws background of a list item.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void DrawItemBackground(  
    Rect position,  
    int index  
)
```

### Parameters

*position*

Type: **Rect**

Total position of list element in GUI.

*index*

Type: **SystemInt32**

Zero-based index of array element.

### Implements

[IReorderableListAdaptorDrawItemBackground\(Rect, Int32\)](#)

## ► Remarks

This method is only used to handle GUI repaint events.

Background of list item spans a slightly larger area than the main interface that is drawn by [DrawItem\(Rect, Int32\)](#) since it is drawn

behind the grab handle.

## ◀ See Also

[Reference](#)

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# SerializedPropertyAdaptorDuplicate Method

Duplicate existing element.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public void Duplicate(  
    int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index of list element.

### Implements

[IReorderableListAdaptorDuplicate\(Int32\)](#)

## ► Remarks

Consider using the **ICloneable** interface to duplicate list elements where appropriate.

## ► See Also

[Reference](#)

**SerializedPropertyAdaptor Class**  
**Rotorz.ReorderableList Namespace**

---

©2013-2015 Rotorz Limited. All rights reserved.

# SerializedPropertyAdaptorEndGUI Method

Occurs after all list items have been drawn.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual void EndGUI()
```

Implements

[IReorderableListAdaptorEndGUI](#)

## ► Remarks

This method is only used to handle GUI repaint events.

## ► See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptor.GetItemHeight Method

Gets height of list item in pixels.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public virtual float GetItemHeight(  
    int index  
)
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index of array element.

### Return Value

Type: **Single**

Measurement in pixels.

### Implements

[IReorderableListAdaptor.GetItemHeight\(Int32\)](#)

## ► See Also

[Reference](#)

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

©2013-2015 Rotorz Limited. All rights reserved.

# SerializedPropertyAdaptorInsert Method

Insert new element at specified index.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public void Insert(  
    int index  
)
```

### Parameters

*index*

Type: [SystemInt32](#)

Zero-based index for list element.

### Implements

[IReorderableListAdaptorInsert\(Int32\)](#)

## ► See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptorMove Method

Move element from source index to destination index.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public void Move(  
    int sourceIndex,  
    int destIndex  
)
```

## Parameters

*sourceIndex*

Type: **SystemInt32**

Zero-based index of source element.

*destIndex*

Type: **SystemInt32**

Zero-based index of destination element.

## Implements

[IReorderableListAdaptorMove\(Int32, Int32\)](#)

## ► See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

---

©2013-2015 Rotorz Limited. All rights reserved.

# SerializedPropertyAdaptorRemove Method

Remove element at specified index.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public void Remove(  
    int index  
)
```

### Parameters

*index*

Type: [SystemInt32](#)

Zero-based index of list element.

### Implements

[IReorderableListAdaptorRemove\(Int32\)](#)

## ► See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptor Properties

The [SerializedPropertyAdaptor](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">arrayProperty</a>	Gets the underlying serialized array property.
	<a href="#">Count</a>	Gets count of elements in list.
	<a href="#">Item</a>	Gets element from list.

[Top](#)

## See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptorarrayProp Property

Gets the underlying serialized array property.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public SerializedProperty arrayProperty { get; }
```

Property Value

Type: **SerializedProperty**

## ► See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptorCount Property

Gets count of elements in list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public int Count { get; }
```

Property Value

Type: [Int32](#)

Implements

[IReorderableListAdaptorCount](#)

## ► See Also

Reference

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

# SerializedPropertyAdaptorItem Property

Gets element from list.

**Namespace:** [Rotorz.ReorderableList](#)

**Assembly:** Editor.ReorderableList (in Editor.ReorderableList.dll)

Version: 0.0.0.0 (0.3.0.0)

## ► Syntax

C#    UnityScript

[Copy](#)

```
public SerializedProperty this[  
    int index  
] { get; }
```

### Parameters

*index*

Type: **SystemInt32**

Zero-based index of element.

### Return Value

Type: **SerializedProperty**

Serialized property wrapper for array element.

## ► See Also

[Reference](#)

[SerializedPropertyAdaptor Class](#)

[Rotorz.ReorderableList Namespace](#)

# Serialized property inside custom inspector

Reorderable list fields can be added to custom inspector interfaces with automatic support for undo/redo when using serialized properties. Serialized properties support native arrays as well as generic lists.

In this example we will implement an editor for the following behaviour class:

C#    UnityScript

Copy

```
using System.Collections.Generic;
using UnityEngine;

public class SomeBehaviour : MonoBehaviour {
    public List<string> wishlist = new List<string>;
}
```

Custom inspectors can be implemented by extending the [Editor Class](#). The serialized property for our "wishlist" field can then be accessed via the serialize object representation of "SomeBehaviour". We can override the method [OnInspectorGUI](#) to present the reorderable list.

C#    UnityScript

Copy

```
using Rotorz.ReorderableList;
using UnityEditor;
using UnityEngine;

[CustomEditor(typeof(SomeBehaviour))]
public class SomeBehaviourEditor : Editor {

    private SerializedProperty _wishlistProperty;

    void OnEnable() {
```

```
        _wishlistProperty = serializedObject.FindP  
    }  
  
    public override void OnInspectorGUI() {  
        serializedObject.Update();  
  
        ReorderableListGUI.ListField(_wishlistProp  
        serializedObject.ApplyModifiedProperties()  
    }  
}
```

---

©2013-2015 Rotorz Limited. All rights reserved.

# Generic list inside editor window

Items from generic lists can be presented using custom item drawers. A custom item drawer is essentially a delegate which is called to draw each list item. The generic list adaptor can be subclassed instead if items of varying heights are needed (see [Custom list adaptor](#)).

## Tip

Consider using [serialized properties](#) instead if undo/redo support is needed.

C#    UnityScript

[Copy](#)

```
using Rotorz.ReorderableList;
using System.Collections.Generic;
using UnityEditor;
using UnityEngine;

public class GenericListWindow : EditorWindow {

    private List<string> _nameList;

    void OnEnable() {
        _nameList = new List<string>();
    }

    void OnGUI() {
        ReorderableListGUI.ListField(_nameList, DrawListItem);
    }

    string DrawListItem(Rect position, string value)
        // Text fields do not like null values!
        if (value == null)
            value = "";
        return EditorGUI.TextField(position, value);
    }
}
```

}



---

©2013-2015 Rotorz Limited. All rights reserved.

# Serialized property inside editor window

The serialized property version of this field can also be used in editor windows. This allows you to take advantage of the automatic undo and redo capabilities which Unity provides.

In this example a reorderable list field is shown when an object containing the following behaviour is selected:

C#    UnityScript

Copy

```
using System.Collections.Generic;
using UnityEngine;

public class SomeBehaviour : MonoBehaviour {
    public List<string> wishlist = new List<string>;
}
```

Before we can interact with the wishlist property we must create an instance of [SerializedObject](#). This is done each time the user selection changes.

C#    UnityScript

Copy

```
using Rotorz.ReorderableList;
using UnityEditor;
using UnityEngine;

public class ArrayPropertyWindow : EditorWindow {

    private SerializedObject _serializedObject;
    private SerializedProperty _wishlistProperty;

    void OnEnable() {
        // Consider selection when window is first
```

```
        OnSelectionChange();
    }

    void OnSelectionChange() {
        // Get editable `SomeBehaviour` objects from Selection
        var filtered = Selection.GetFiltered(typeo...
        if (filtered.Length == 0) {
            _serializedObject = null;
            _wishlistProperty = null;
        }
        else {
            // Let's work with the first filtered
            _serializedObject = new SerializedObject(filtered[0]);
            _wishlistProperty = _serializedObject.FindProperty("...
        }
    }

    Repaint();
}

void OnGUI() {
    if (_serializedObject == null)
        return;
    _serializedObject.Update();

    ReorderableListGUI.ListField(_wishlistProp...
    _serializedObject.ApplyModifiedProperties();
}

}
```

# Customize appearance of list field

Style of list container, add button and remove buttons can be customized by providing custom styles. This example demonstrates a custom inspector with custom styles which are based upon the default styles.

## Tip

Another option is to subclass `ReorderableListControl` and initialise custom styles there instead. A subclass can override other behaviour such as providing custom context menu items.

C#    UnityScript

Copy

```
using Rotorz.ReorderableList;
using UnityEditor;
using UnityEngine;

[CustomEditor(typeof(SomeBehaviour))]
class SomeBehaviourEditor : Editor {

    // Custom instance of reorderable list control
    private ReorderableListControl _listControl;
    // Serialized property adaptor for wishlist.
    private SerializedListAdapter _wishlistAdaptor

    void OnEnable() {
        // Prepare custom styles as needed.
        var style = new GUIStyle(ReorderableListStyle);
        style.normal.background = AssetDatabase.LoadAssetAtPath<Texture2D>(
```

// Assign custom style to instance of list

```
_listControl = new ReorderableListControl(
    _listControl.ContainerStyle = style;
```

// Create adaptor for wishlist using seria

```
        var wishlist = serializedObject.FindProperty("wishlist");
        _wishlistAdaptor = new SerializedListAdapter(wishlist);
    }

    public override void OnInspectorGUI() {
        listControl.Draw(_wishlistAdaptor);
    }
}
```

---

©2013-2015 Rotorz Limited. All rights reserved.

# Detect changes within list field

The reorderable list field watches for changes surrounding individual items allowing you to pinpoint the item which was actually modified. Any changes made outside of list item drawers are considered changes made by the list control itself.

The property `GUI.changed` is set to true if changes are made using the list control. If a list item drawer reports changes then the property `ReorderableListGUI.indexOfChangedItem` is set to the zero-based index of that item. A value of -1 indicates that changes were reported by the list control itself.

C# UnityScript

Copy

```
// Begin checking for changes to `GUI.changed`.
EditorGUI.BeginChangeCheck();

ReorderableListGUI.ListField(wishlist);

// Were any changes made to the state of `GUI.changed`?
if (EditorGUI.EndChangeCheck()) {
    // Determine whether changes were made to a specific item.
    if (ReorderableListGUI.IndexOfChangedItem != -1) {
        // We know the index of the item which was changed.
    }
    else {
        // Changes were made outside of an item drawer.
        // for example, an item was added, removed
    }
}
```

# Custom list adaptor

This example demonstrates how to implement a custom list adaptor which can contain caption items which cannot be dragged or removed. List items can however be dragged around the stationary caption items.



Since in this example we are working with a straightforward list of strings it makes sense to subclass [GenericListAdapterT](#).

A simple naming convention will be assumed to differentiate between regular entries and captions. String entries which are enclosed between curly brackets will be presented differently and cannot be reordered or removed using the list control interface. A simple function can be created to test whether a list entry is a caption like `IsCaption` below.

C#    UnityScript

[Copy](#)

```
using Rotorz.ReorderableList;
using System.Collections.Generic;
using UnityEditor;

public class SpecialAdaptor : GenericListAdapter<string> {
    public SpecialAdaptor(List<string> list, ReorderableList listControl)
        : base(list, itemDrawer, itemHeight) {
    }
}
```

```

public override void DrawItem(Rect position, int index, ref Rect rect) {
    string item = this[index];
    if (IsCaption(item))
        GUI.Label(position, item.Substring(1, item.Length - 1));
    else
        base.DrawItem(position, index);
}

public override bool CanDrag(int index) {
    return !IsCaption(this[index]);
}
public override bool CanRemove(int index) {
    return !IsCaption(this[index]);
}

public override float GetItemHeight(int index) {
    return IsCaption(this[index]) ? 28 : fixedHeight;
}

private bool IsCaption(string item) {
    return item != null && item.Length > 0
        && item[0] == '{' && item[item.Length - 1] == '}';
}

```

Our custom adaptor must then be instantiated before it can be used; this instance can be cached if desired. The adaptor instance can then be passed to [ReorderableListGUILayout](#) or even a custom list control using [ReorderableListControlDraw](#).

C#    UnityScript

[Copy](#)

```
var adaptor = new SpecialAdaptor(list, itemDrawer,
ReorderableListGUI.ListField(adaptor);
```



# Customize context menu

This example demonstrates how to add to, or even replace entirely, the list field's context menu.

First we need to implement a subclass of `ReorderableListControl` so that we can override the default context menu and add some custom items.

C#    UnityScript

Copy

```
using Rotorz.ReorderableList;
using UnityEditor;

public class CustomContextMenuList : ReorderableListControl
{
    private static readonly GUIContent s_MenuItem1 = new GUIContent("Custom Item 1");
    private static readonly GUIContent s_MenuItem2 = new GUIContent("Custom Item 2");

    protected override void AddItemsToMenu(GenericMenu menu)
    {
        // Remove if default menu items are not wanted
        base.AddItemsToMenu(menu, itemIndex, adapter);

        menu.AddSeparator(" ");

        // Custom menu item the usual way:
        menu.AddItem(s_MenuItem1, false, () => Debug.Log("Custom Item 1"));
        // Or... implement as command:
        menu.AddItem(s_MenuItem2, false, defaultCommand);
    }

    protected override bool HandleCommand(string commandName)
    {
        // Remove if default commands are not wanted
        if (base.HandleCommand(commandName, itemIndex))
            return true;

        // Place custom command handler here...
    }
}
```

```
        switch (commandName) {
            case "MenuItem2":
                Debug.Log("You selected menu item :");
                return true;
        }

        return false;
    }

}
```

In order to use our custom reorderable list we will also need to instantiate an adaptor for our list. We can cache this adaptor alongside our custom list control instance.

C#    UnityScript

[Copy](#)

```
private SerializedProperty _someListProperty;

private CustomContextMenuList _customListControl;
private IReorderableListAdapter _someListAdapter;

void OnEnable() {
    _someListProperty = serializedObject.FindProperty("list");

    _customListControl = new CustomContextMenuList();
    _someListAdapter = new SerializedPropertyAdapter(_someListProperty);
}

public override void OnInspectorGUI() {
    serializedObject.Update();

    _customListControl.Draw(_someListAdapter);

    serializedObject.ApplyModifiedProperties();
}
```

---

©2013-2015 Rotorz Limited. All rights reserved.

# Subscribing to item inserted and removing events

This example demonstrates how to subscribe to events by creating a local instance of the list control. It is also necessary to instantiate the appropriate list adaptor which can be cached alongside list control instance.

For this example we will subscribe to item added and removing events so that we can echo these to the Unity console.

C#    UnityScript

[Copy](#)

```
using Rotorz.ReorderableList;
using System.Collections.Generic;
using UnityEditor;

public class ExampleWindow : EditorWindow {

    private ReorderableListControl _listControl;
    private IReorderableListAdapter _listAdaptor;

    private List<string> _someList = new List<string>();

    private void OnEnable() {
        // Create list control and optionally pass
        // some list to it.
        _listControl = new ReorderableListControl(_someList);

        // Subscribe to events for item insertion / removal.
        _listControl.ItemInserted += OnItemInserted;
        _listControl.ItemRemoving += OnItemRemoving;

        // Create adaptor for example list.
        _listAdaptor = new GenericListAdapter(_someList);
    }

}
```

```
private void OnDisable() {
    // Unsubscribe from events, good practice.
    if (_listControl != null) {
        _listControl.ItemInserted -= OnItemInserted;
        _listControl.ItemRemoving -= OnItemRemoving;
    }
}

private void OnItemInserted(object sender, ItemChangedEventArgs args) {
    string item = _someList[args.ItemIndex];
    if (args.WasDuplicated)
        Debug.Log("Duplicated: " + item);
    else
        Debug.Log("Inserted: " + item);
}

private void OnItemRemoving(object sender, ItemRemovingEventArgs args) {
    string item = _someList[args.ItemIndex];
    Debug.Log("Removing: " + item);

    // You can cancel item removal at this stage
    if (item == "Keep Me!")
        args.Cancel = true;
}

private void OnGUI() {
    // Draw layout version of reorderable list
    _listControl.Draw(_listAdaptor);

    // OR

    // Draw absolute version of reorderable list
    Rect position = default(Rect);
    position.x = 100;
    position.y = 100;
    position.width = 200;
    position.height = _listControl.CalculateHeight();
    _listControl.Draw(position, _listAdaptor);
}
```

}

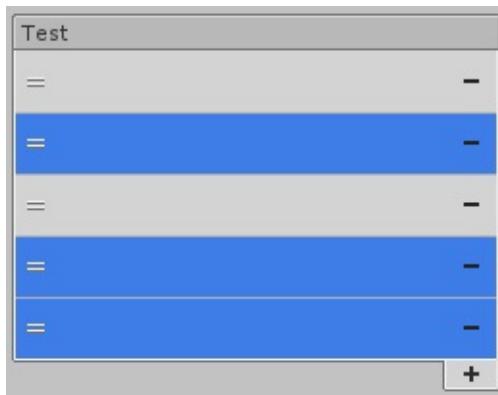


---

©2013-2015 Rotorz Limited. All rights reserved.

# Item selection with a custom adaptor

Item selection can be added to a reorderable list control by creating a custom reorderable list adaptor. Selection state can be efficiently represented using a hash set collection.



C#    UnityScript

[Copy](#)

```
using Rotorz.ReorderableList;
using System.Collections.Generic;
using UnityEngine;

public class SelectableItemAdaptor : GenericListAdapter<string> {

    private HashSet<int> _selectedIndices = new HashSet<int>();

    public SelectableItemAdaptor(List<string> list)
        : base(list, null, itemHeight) {
    }

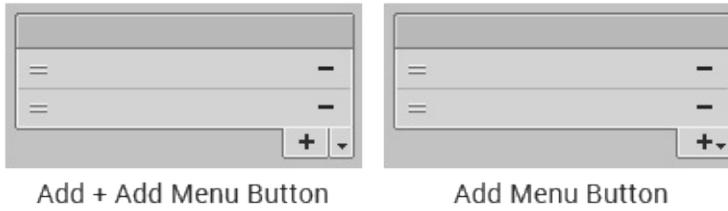
    public override void DrawItemBackground(Rect position) {
        if (_selectedIndices.Contains(index))
            ReorderableListStyles.SelectedItem.Draw(position);
    }
}
```

```
public override void DrawItem(Rect position, int controlID = GUIUtility.GetControlID(FocusType.ListView))
{
    switch (Event.current.GetTypeForControl(controlID))
    {
        case EventType.MouseDown:
            if (Event.current.button == 0 && position.Contains(Event.current.mousePosition))
            {
                if (Event.current.control)
                {
                    // Toggle selection of this item
                    if (_selectedIndices.Contains(index))
                        _selectedIndices.Remove(index);
                    else
                        _selectedIndices.Add(index);
                }
                else
                {
                    // Deselect all other items
                    _selectedIndices.Clear();
                    _selectedIndices.Add(index);
                }
                Event.current.Use();
            }
            break;

        case EventType.Repaint:
            GUI.skin.label.Draw(position, this);
            break;
    }
}
```

# Adding the drop-down add menu

The drop-down add menu button is automatically displayed when there is at least one subscriber to the **AddMenuClicked** event. The presentation of the button varies depending upon whether the regular add button is also shown.



The drop-down add button is a general purpose button with no default behavior which can be used to display a menu or a drop-down window.

In this example a simple menu is constructed and shown upon clicking the add menu button:

C#    UnityScript

[Copy](#)

```
using Rotorz.ReorderableList;
using System.Collections.Generic;
using UnityEditor;
using UnityEngine;

public class ExampleWindow : EditorWindow {

    private ReorderableListControl _listControl;
    private IReorderableListAdapter _listAdaptor;

    private List<string> _someList = new List<string>();

    private void OnEnable() {
        // Create list control and pass flag into it.
        // regular add button is not displayed.
        _listControl = new ReorderableListControl(_someList, true);
        _listControl.OnAddItem += () => {
            _someList.Add("New Item");
        };
        _listControl.OnMoveItem += (int fromIndex, int toIndex) => {
            string item = _someList[fromIndex];
            _someList.RemoveAt(fromIndex);
            _someList.Insert(toIndex, item);
        };
        _listControl.OnRemoveItem += (int index) => {
            _someList.RemoveAt(index);
        };
    }
}
```

```

        // Subscribe to event for when add menu button is clicked.
        // also indicate that the add menu button has been clicked.
        _listControl.AddMenuClicked += OnAddMenuItemClicked;

        // Create adaptor for example list.
        _listAdaptor = new GenericListAdaptor<string>();
    }

    private void OnDisable() {
        // Unsubscribe from event, good practice.
        if (_listControl != null)
            _listControl.AddMenuClicked -= OnAddMenuItemClicked;
    }

    private void OnAddMenuItemClicked(object sender, EventArgs args) {
        var menu = new GenericMenu();
        menu.AddItem(new GUIContent("Tree"), false);
        menu.AddItem(new GUIContent("Bush"), false);
        menu.AddItem(new GUIContent("Grass"), false);
        menu.DropDown(args.ButtonPosition);
    }

    private void OnSelectAddMenuItem(object userData) {
        Debug.Log(userData);
    }

    private void OnGUI() {
        // Draw layout version of reorderable list
        _listControl.Draw(_listAdaptor);
    }
}

```

# Populating the drop-down add menu with types

Utility functionality is provided to assist when building menus that contain a number of addable element types that implement some interface or base type; this is referred to as the element contract type.

Let's begin by defining the base class that each of our elements must be derived from:

C#    UnityScript

[Copy](#)

```
// ExampleNode.cs
using UnityEngine;

public abstract class ExampleNode : ScriptableObject {
    [SerializeField]
    private string _displayName;

    public string DisplayName {
        get { return _displayName; }
        set { _displayName = value; }
    }

}
```

We then need some sort of container wherein our node instances will be stored. In the case of this example this will be another custom ScriptableObject implementation; although the same principle can also be applied to a collection type.

C#    UnityScript

[Copy](#)

```
// ExampleGraph.cs
using System.Collections.Generic;
```

```
using UnityEngine;

public abstract class ExampleGraph : ScriptableObject
{
    [SerializeField]
    private List<ExampleNode> _nodes = new List<ExampleNode>();

    public void AddNode(ExampleNode node)
    {
        _nodes.Add(node);
    }

}
```

We will also need to implement at least one type of node so that the add menu will contain at least one type to select from!

C# UnityScript

[Copy](#)

```
// NodeTypeA.cs
public class NodeTypeA : ExampleNode {}

// NodeTypeB.cs
public class NodeTypeB : ExampleNode {}
```

An **IElementAdderTContext** implementation is also needed since this defines how nodes are to be created and how they are to be associated with their context object. In the case of this example the context object will be an instance of ExampleGraph.

C# UnityScript

[Copy](#)

```
using Rotorz.ReorderableList;
using System;
using UnityEngine;

public class ExampleNodeElementAdder : IElementAdder
```

```

public ExampleNodeElementAdder(ExampleGraph graph)
    Object = graph;
}

public ExampleGraph Object { get; private set; }

public bool CanAddElement(Type type) {
    return true;
}

public object AddElement(Type type) {
    var node = (ExampleNode)ScriptableObject.C
Object.AddNode(node);
    return node;
}

}

```

The drop-down add menu can then be defined like shown below where we make use of an adder element menu builder to populate the menu with the relevant element types:

C#    UnityScript    [Copy](#)

```

using Rotorz.ReorderableList;
using UnityEditor;
using UnityEngine;

[CustomEditor(typeof(ExampleGraph))]
public class ExampleGraphEditor : Editor {

    private ReorderableListControl _listControl;
    private IReorderableListAdapter _listAdaptor;

    private void OnEnable() {
        // Create list control and pass flag into it
        // regular add button is not displayed.
        _listControl = new ReorderableListControl()

```

```

    // Subscribe to event for when add menu button is clicked.
    // also indicate that the add menu button is active.
    _listControl.AddMenuClicked += OnAddMenuClicked;

    // Create adaptor for example list.
    var nodesProperty = serializedObject.FindProperty("nodes");
    _listAdaptor = new SerializedPropertyAdapter(nodesProperty);
}

private void OnDisable() {
    // Unsubscribe from event, good practice.
    if (_listControl != null)
        _listControl.AddMenuClicked -= OnAddMenuClicked;
}

private void OnAddMenuClicked(object sender, EventArgs args) {
    var graph = target as ExampleGraph;
    var elementAdder = new ExampleNodeElementAdder(graph);

    var builder = ElementAdderMenuBuilder.ForElementAdder(elementAdder);
    builder.SetElementAdder(elementAdder);

    var menu = builder.GetMenu();
    menu.DropDown(args.ButtonPosition);
}

private void OnGUI() {
    // Draw layout version of reorderable list
    _listControl.Draw(_listAdaptor);
}
}

```

With this approach custom commands can also be included by adding them directly using the menu builder.

Adder menus can also be extended with custom commands without needing to directly interact with the menu builder. This can be achieved by annotating custom command implementations with an attribute

which defines the context in which the command will be included:

C#    UnityScript

[Copy](#)

```
using Rotorz.ReorderableList;
using UnityEngine;

[ElementAdderMenuCommand(typeof(ExampleNode))]
public class SpecialCommand : IElementAdderCommand {

    public SpecialCommand() {
        Content = new GUIContent("Special Command")
    }

    public GUIContent Content { get; private set; }

    public bool CanExecute(IElementAdder<ExampleGraph> adder) {
        return true;
    }

    public void Execute(IElementAdder<ExampleGraph> adder) {
        // Execute some custom command here!
        // Such as bulk adding nodes!
    }
}
```