ObservableList

Events Event handlers for add, remove and clear events are available on ObservableList objects.

```
private void OnEnable()
{

    list.AddEvent += OnAddEvent;
    list.ClearEvent += OnClearEvent;
    list.RemoveEvent += OnRemoveEvent;

}
```

When adding a handler to any event, make sure and remove it when the script it is associated with is disabled.

```
private void OnDisable()
{

    list.AddEvent -= OnAddEvent;
    list.ClearEvent -= OnClearEvent;
    list.RemoveEvent -= OnRemoveEvent;

}
```

Count Gets the number of elements contained in the ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

Debug.Log(list.Count);
```

IsReadOnly Gets a value indicating whether the ObservableList is read-only.

```
ObservableList<int> list = new ObservableList<int>();

Debug.Log(list.IsReadOnly);
```

Add Adds an item to the end of the ObservableList.

```
ObservableList<int> list = new ObservableList<int>();

list.Add(1);
```

AddRange Adds the items of a List to the end of the ObservableList.

```
ObservableList<int> list = new ObservableList<int>();

list.AddRange(new List<int> { 1, 2, 3 });
ObservableList<int> list = new ObservableList<int>();

list.AddRange(new ObservableList<int> { 1, 2, 3 });
```

Clear Removes all items from the ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

list.Clear();
```

Contains Determines whether an item is in the ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

Debug.Log(list.Contains(2));
```

CopyTo Copies all items in the ObservableList to the array starting at the `arrayIndex`.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

int[] array = new int[10];

list.CopyTo(array, 0);
```

GetRange Creates a shallow copy of a range of items in the source ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

ObservableList<int> newList = list.GetRange(1, 2);
```

IndexOf Searches for the specified item and returns the zero-based index of the first occurrence within the entire ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

Debug.Log(list.IndexOf(2));
```

Insert Inserts an item into the ObservableList at the specified index.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

list.Insert(1, 6);
```

InsertRange Inserts the items of a List into the ObservableList at the specified index.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

list.InsertRange(0, new List<int> { -1, 0 });
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

list.InsertRange(0, new ObservableList<int> { -1, 0 });
```

Pop Removes the last item from an ObservableList and returns that item.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

int lastItemInList = list.Pop();
```

Random Returns a random item from an ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

int randomItemFromList = list.Random();
```

Remove Removes the first occurrence of a specific item from the ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

list.Remove(1);
```

RemoveAt Removes the item at the specified index of the ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

list.RemoveAt(0);
```

RemoveRange Removes a range of items from the ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

list.RemoveRange(1, 2);
```

Shift Removes the first item from an ObservableList and returns that item.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

int firstItemInList = list.Shift();
```

Shuffle Creates a new copy of an ObservableList and shuffles the items.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

ObservableList<int> shuffledList = numberRange.Shuffle();
```

Shuffle with a specific seed.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

ObservableList<int> shuffledList = numberRange.Shuffle(10);
```

Slice Returns a shallow copy of a portion of an ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

ObservableList<int> slicedItemsList = numberRange.Slice(2);
```

Splice Removes and returns a shallow copy of a portion of an ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

ObservableList<int> removedItemsList = numberRange.Splice(1, 2);
```

ToList Creates a List with the values from an ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

List<int> newList = list.ToList();
```

Unshift Adds a range of items to the beginning of an ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

list.Unshift(new List<int> { -1, 0 });
```

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

list.Unshift(new ObservableList<int> { -1, 0 });
```

Adds an item to the beginning of an ObservableList.

```
ObservableList<int> list = new ObservableList<int> { 1, 2, 3, 4, 5 };

list.Unshift(0);
```