Events Plus 2.0 Documentation

What is it?

Events Plus is a powerful event management tool for the Unity Engine. It has been designed as a complete replacement of the stock UnityEvent system and includes several added features and performance improvements. Like UnityEvents, Events Plus enables developers to avoid otherwise cumbersome event wiring and instead manage it all solely within the editor window.

Core Features

- Automatic event wiring
- Supports every Unity variable type (Vector3, AnimationCurve, etc.)
- Allows up to 10 event arguments
- Expandable and re-orderable inspector drawers
- Optimized for cross-platform with up to 1000% faster speeds over stock UnityEvents
- Supports both direct events and preconfigured argument calls to all public variables, properties and methods
- Supports methods with return types and function overloading
- Source code included and fully-documented

Quick Start

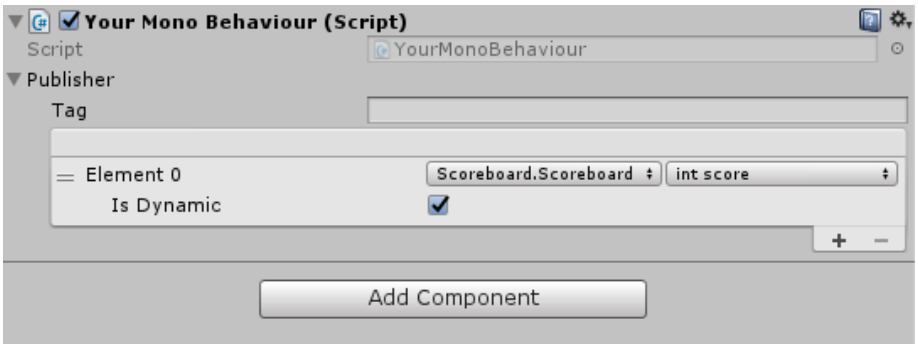
Events Plus setup is similar to UnityEvents, but requires an extra line of code for initialization that can be inserted into the *Awake()* function. This is necessary for registering everything to the event system and avoiding any possible memory leaks.

```
[Serializable]
public class PublisherInt : Publisher<int>
{
}
```

```
public class YourMonoBehaviour : MonoBehaviour
{
	public PublisherInt publisher;

	public void Awake()
	{
		publisher.initialize();
	}

	public void Start()
	{
		publisher.publish( 5 );
	}
}
```



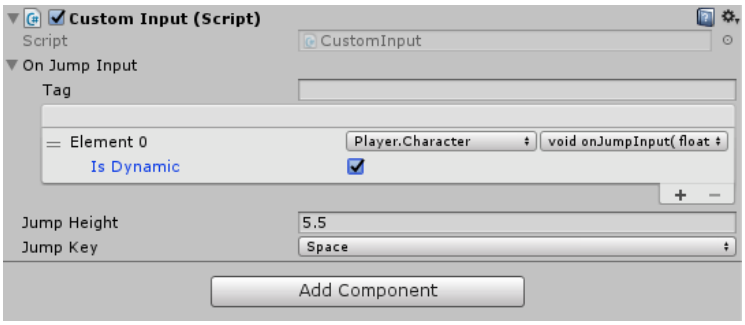
Publisher

A Publisher works just like a standard UnityEvent. It contains a list of desired event calls that can be setup to be either dynamically invoked (i.e., controlled via code) or allow predefined arguments to be set directly in the inspector.

Dynamic Calls

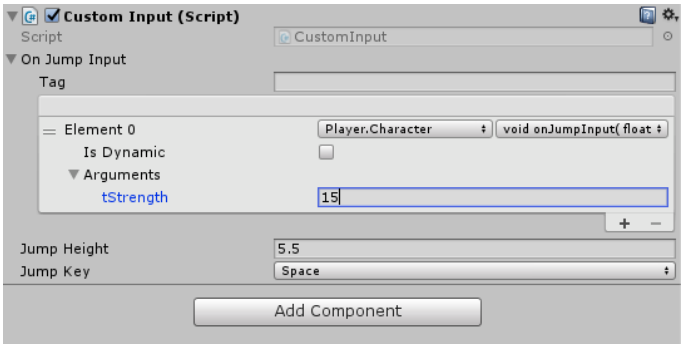
If a call is marked as dynamic, the source code that fires the *publish()* function will control what arguments get passed to the call:

```
publisher.publish( 1, true );
```



Preconfigured Calls

If a call is not marked as dynamic, arguments can be set directly within the inspector and will be passed into the call's function whenever the *publish()* function is fired.

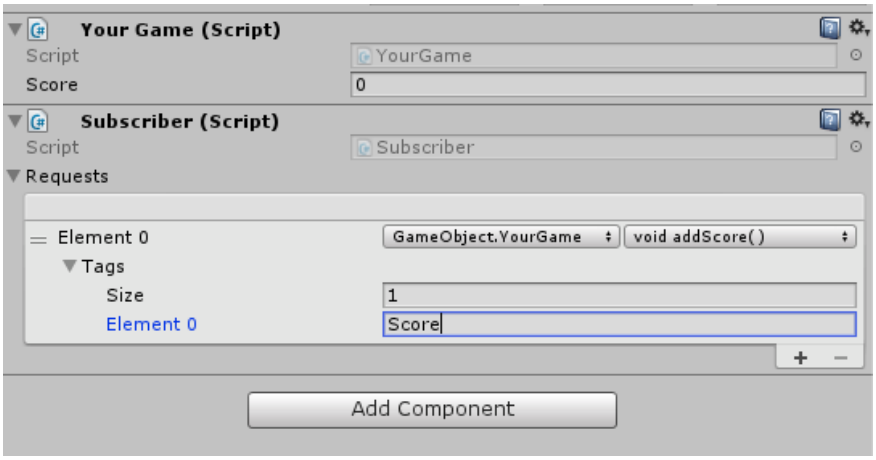


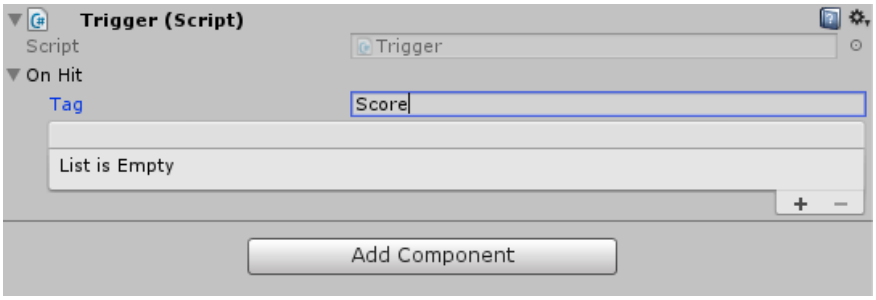
Subscriber

A Subscriber is an optional MonoBehaviour that can used to automatically wire up events globally. This is especially useful for spawned objects that need to be wired up.

Requests

Every Subscriber contains a list of event requests that operate similar to the Publisher's list of calls. Each request contains a tag list that allows them to automatically wire up to any Publishers with a matching tag name. This will occur at the start of gameplay.





Release Notes

1.1

- Updated to the latest 2017 Unity build
- Fixed specific crash issue with IOS
- Replaced the “channel” integer with a string “tag” name for request tracking
- Added an early version of a “Settings” panel for method filtering
- Included an example that utilizes a spawned Singleton instance

2.0

Note: This version has undergone significant changes for the 2018 version of Unity and will cause work from the older versions to break!

- Overhauled and refactored source code for usability and performance improvements
- Increased the number of arguments supported from 6 to 10
- Fixed a rare bug that caused Publishers to not work when treated as elements in an array
- Added drop-down support for member filtering inside of the “Settings” panel
- Fixed bug that caused multiple MonoBehaviours of the same type and applied to the same object to not show up in the inspector drop-downs
- Added extra guards against potential memory leaks and added thread-safety