
KennethDevelops
EventManager

Thank you for purchasing EventManager!

EventManager is a tool that helps you ensure communication between the elements on your game logic. In other words, it makes it easier to **notify key events to your GameObjects**.

For example, **when the game is over**, you want to **notify every enemy to stop attacking** the player. With EventManager, you can simply **trigger the event** "OnGameOver" and **every element in the game that is subscribed** to this event (in this case, the enemies) **will be notified** and react accordingly.

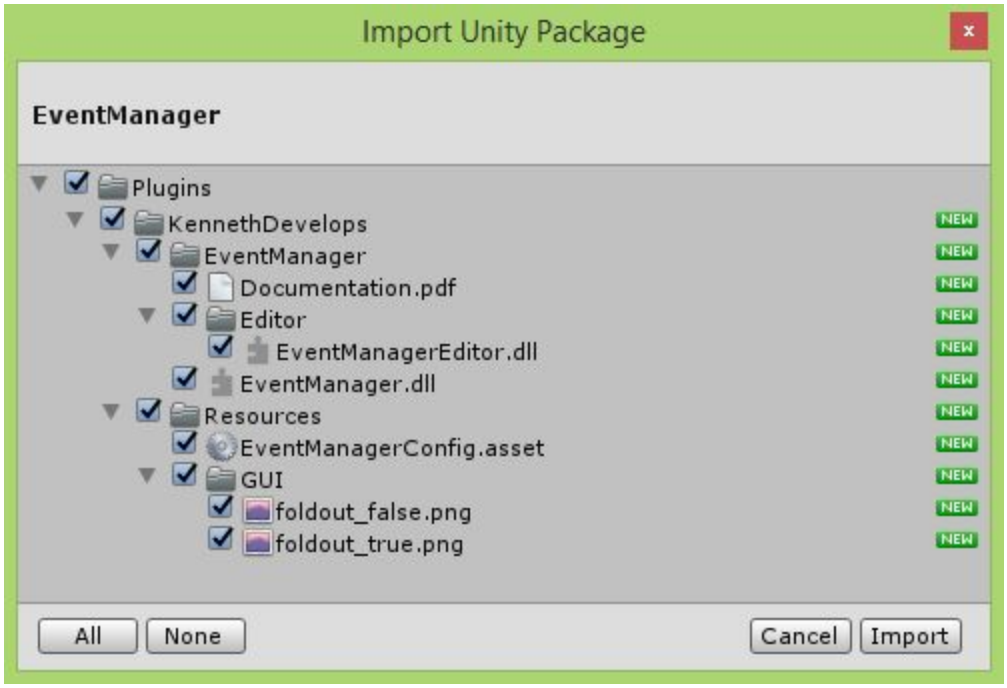
This document intends to serve as a guide to use and master this Asset Package. If possible, please use the online version of this document instead, it may often be updated.

Click here to go to the online version

Table of Contents

Getting Started

Just after purchasing EventManager in the Asset Store you should be prompted to download and import the Asset and then the following dialog will appear:



It is important that the location of the "KennethDevelops" folder remains inside the "Assets/Plugins" folder. Otherwise, the asset may not work properly.

After the import is done, you'll see the Unity Editor loading the asset package for a few seconds and then you'll be ready to begin using EventManager!

Check the other pages in this wiki to see examples about how to use the asset properly.

Support

If you have encountered any issues with the product please fill this form: https://goo.gl/forms/k2bxEeoUS5XkfZKo2

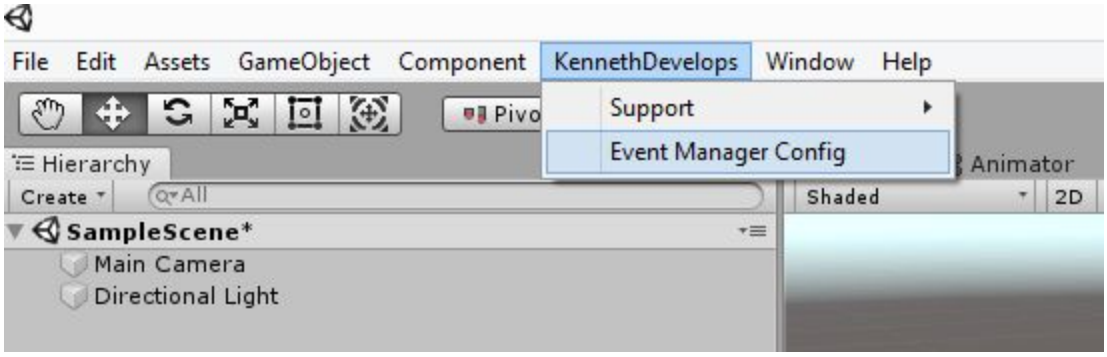
Please be kind enough to explain the issue as best as you can. Images, gifs and/or video will also be appreciated if you think it could help understand the problem.

For any other comment on this wiki or the product itself, feedback or any other kind of suggestions you may have, you can fill this form: https://goo.gl/forms/Bltw82mtyn6pDobf1

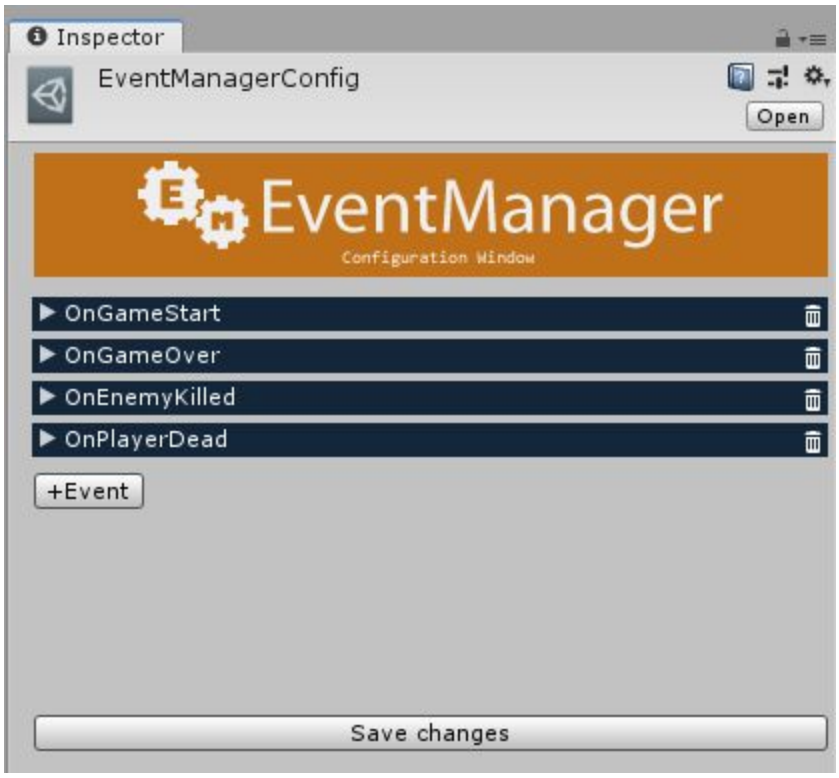
Configuration

Events

To start using the EventManager, first you have to configure your events. To do that, you can go to KennethDevelops->EventManager Config



And then the Inspector will show the configuration window. Here, you will immediately see all your configured events.



Adding Events

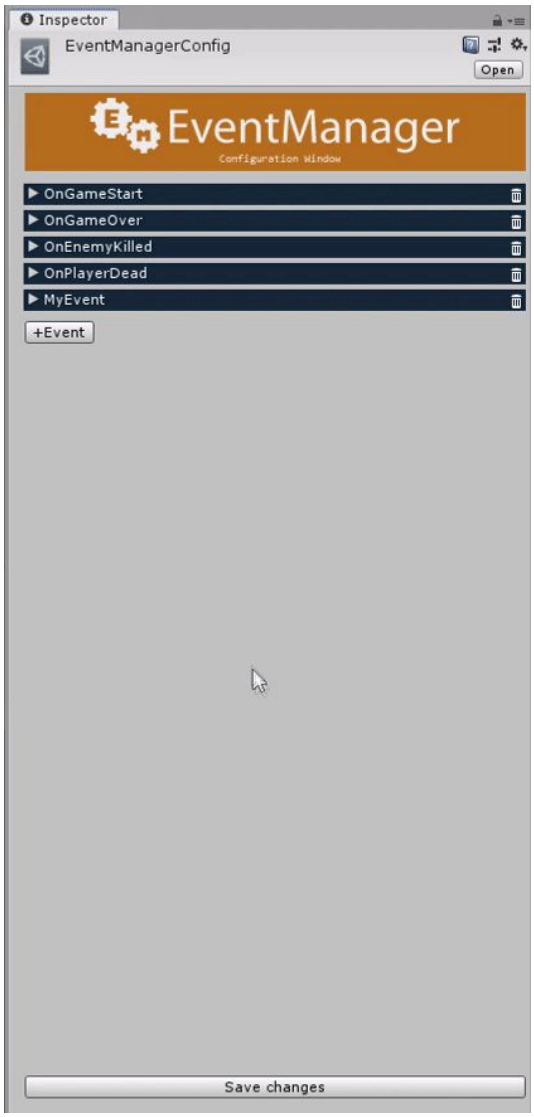
To add an event, simply open the configuration window and press the "+Event" button below the list of events.

Removing Events

To remove an event, simply open the configuration window and press the "-" button at the right side of each event name.

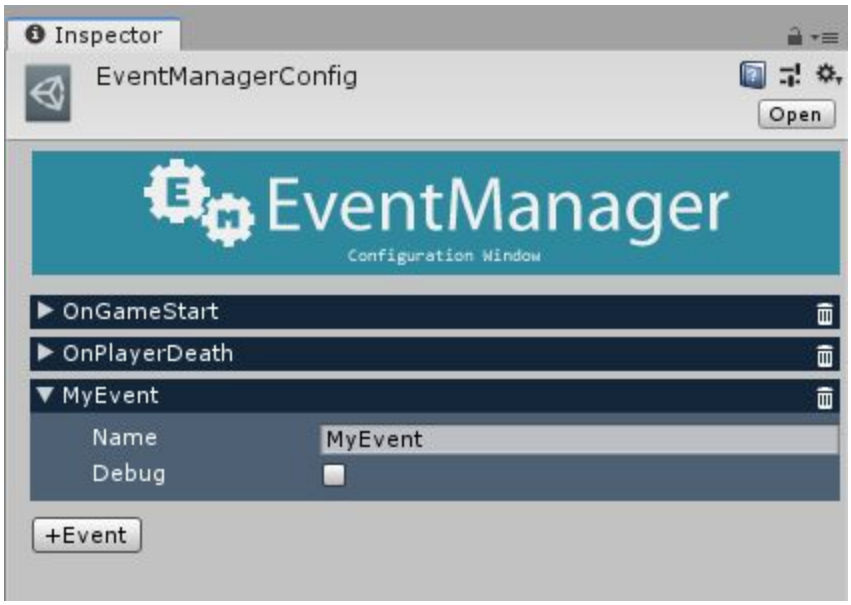
Saving Changes

To save all changes, simply press the "Save changes" button at the bottom of the configuration window. Be sure to do this after every change. Some changes will seem to remain if you don't press this button, even during playmode, but after you restart Unity they will be gone unless you do.



Triggering a DefinedEvent

In this tutorial, we're going to trigger the event "MyEvent" defined here:



First, we need to import the namespace:

```
using KennethDevelops.Events;
```

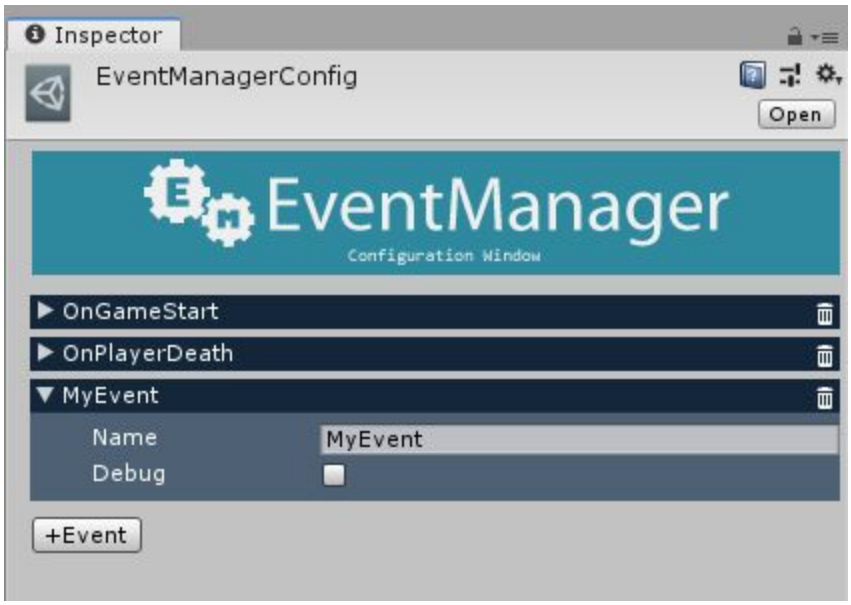
And now, to trigger this event we simply type:

```
new DefinedEvent("MyEvent").Trigger();
```

As easy as that.

Subscribing to a DefinedEvent

In this tutorial, we're going to subscribe to the event "MyEvent" defined here:



First, we need to import the namespace:

```
using KennethDevelops.Events;
```

To subscribe to this event, we simply type:

```
void Start(){
    EventManager.SubscribeToEvent("MyEvent", OnMyEvent);
}

private void OnMyEvent(DefinedEvent definedEvent){

}
```

As we can see, we called the method SubscribeToEvent of the static class EventManager, passed the eventId and then the method that will be executed when our event is triggered.

Unsubscribe to DefinedEvent

To unsubscribe to a DefinedEvent, you simply type:

```
EventManager.UnsubscribeToEvent("eventId", OnEvent);
```

The first string being the event's ID or event's name, and the second being the Callback (the method that you previously subscribed to this event).

For the example case we gave at the beginning of this page, it would be:

```
EventManager.UnsubscribeToEvent("MyEvent", OnMyEvent);
```