System Features – Events and Triggers

# Introduction

In this part the concept and idea of both events and triggers. We will go through their intention and how to use them when using the engine, furthermore a couple of example will be shown in order to give the general idea of what they can be used for.

# Concept

In the natural world all actions have a reaction, these reactions could be thought of as events meant to trigger when such actions are performed. Thus an engine for modeling a virtual environment must provide as many features as those of the real world environment.

## Events

To be clear an Event when speaking in the context of this engine is the occurrence of something, for instance an Event could be that “an Agent has moved”, or “an Agent has picked up an item”. Furthermore an event also has the duty of providing necessary information for the listener giving the listener a correct idea of the meaning behind an event. Going back to the Event of an Agent that has moved, in this case it necessary to provide the listener**(FOOTNOTE: Listeners refer to the object which is listening to the occurrence of an event, with the intent of reacting to it)** with information of which direction the agent moved, where its starting position is and how far it has moved. Since a listener might be operating in a different thread, the listener is completely dependent on this information as it might no longer be retrievable at the time the event is being analyzed. For instance if an agent moved and then was killed and removed from the world, its position would no longer be stored in the world. As such the listener would have no way to determine where the move had ended if not provided in the event.

## Triggers

Triggers in our engine are the means to which listeners gain access to events. A trigger in our engine is the combination of three different parts.

1. Events
2. Condition
3. Action

**The events** are what the Trigger is listening for these can be any type of event, furthermore a trigger can be registered to any number of events. But only one event is required to “trigger” a Trigger for instance if a Trigger is listening on the event “10 seconds passed” and the event “Agent has moved”, then if either of the event has occurred the Trigger will be “triggered”. However it will be triggered each and every time such event has occurred and is not limited to just one occurrence.

**The Condition** is a built in predicate for the trigger to check if it is willing to respond to the event. If the condition is satisfied the trigger’s action is fired. A condition should only be used in cases that is not covered by another event. For instance say you have the event “An agent has moved”, to check if the agent in question is the one you are concerned which we will name A. In order to accommodate this then the condition would be:

Is Agent that has moved, the agent A.

As we can see the condition narrows the range of events that are responded to at the cost of added calculations. In this case it would be much better to use the event “Agent A has moved”, this purely an example as events should of course not be tied to specific objects.

**The Action** of a trigger is meant as the one that performs the work, it is a mere method which is executed once an event has been raised and the condition is satisfied. For instance if a trigger is meant to write a message when a specific event has occurred then this is where the action of writing such message should be placed.

### Entities and EventManager

For triggers to become part of the engine it is required that the trigger is registered to the engine, however it is crucial what one registers the trigger to. A trigger can be registered to either a specific entity or the EventManager. A Trigger registered to the Eventmanager will be triggered each an Event that it is listening to is fired. However a Trigger registered to a specific entity, will only be informed of events raised on the specific entity instead of when the event is raised for every single entity.

An example of this would be: assume you have a Trigger T1 with the event “An agent has moved”, and T is registered to Agent A, also in this example you have a Trigger T2 with same event as T1 but this trigger is registered to the EventManager. To give a complete picture also assume there is an Agent B which has no Triggers registered to it.

This provides us with two scenarios:

1. Agent A is moved:  
   In this case both Trigger T1 and T2 is triggered, since T1 listens on Agent A and T2 listens to any Agent moving.
2. Agent B is moved:  
    In this case only Trigger T2 is triggered, for the reasons stated above.

# Example of making and using an Event

Let’s assume one was to make an event which was fired each time an agent had moved, let us name this Event: AgentMoved.

First make a class extending the XmasEvent class



Then add all the necessary data fields on the newly created event class.



To utilize the newly created event it must be raised when appropriate. In this case the appropriate place would be to raise it during a move action.

In this action after the movement of the agent had been performed the method RaiseEvent would need to be called on the Agent Entity itself that is being moved.

# Summery

Events are what provides the engine flexibility and allows making reactions to others actions, events are designed for easy use and are meant to be used as much as possible. Triggers are used as a way to interface with the Events and they are the only way to connect an object to the event it wishes to listen to.