Event Dispatcher System

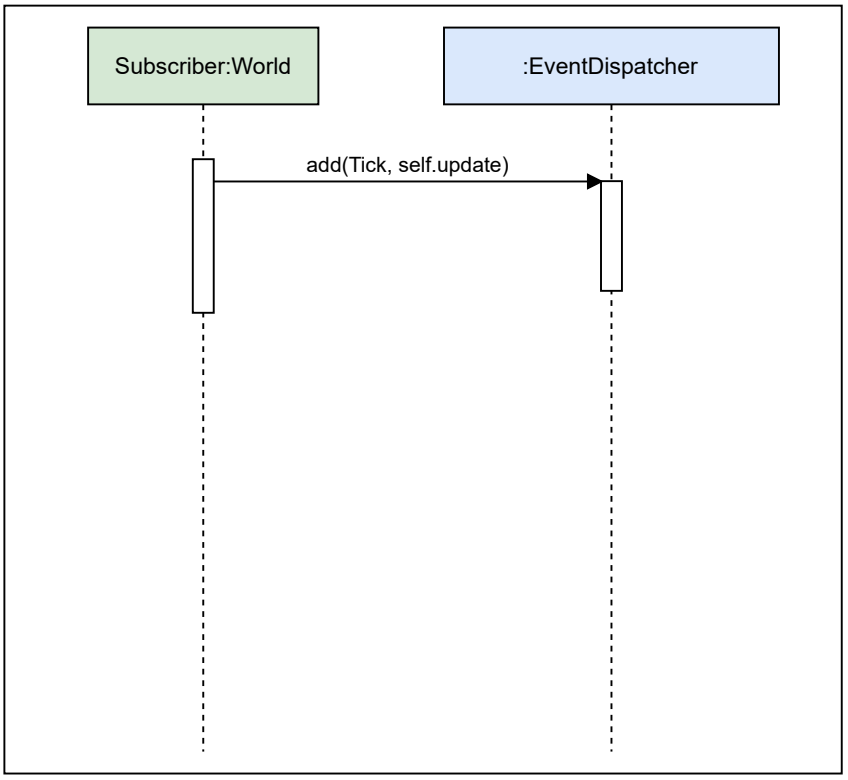
This is NOT publish-subscribe pattern, instead is the observer pattern.

Subscription

To subscribe an object to an event, this will pass to the EventDispatcher two parameters.
First, the event class that it wants to subscribe to.
Second, the method that it wants to recive the event.

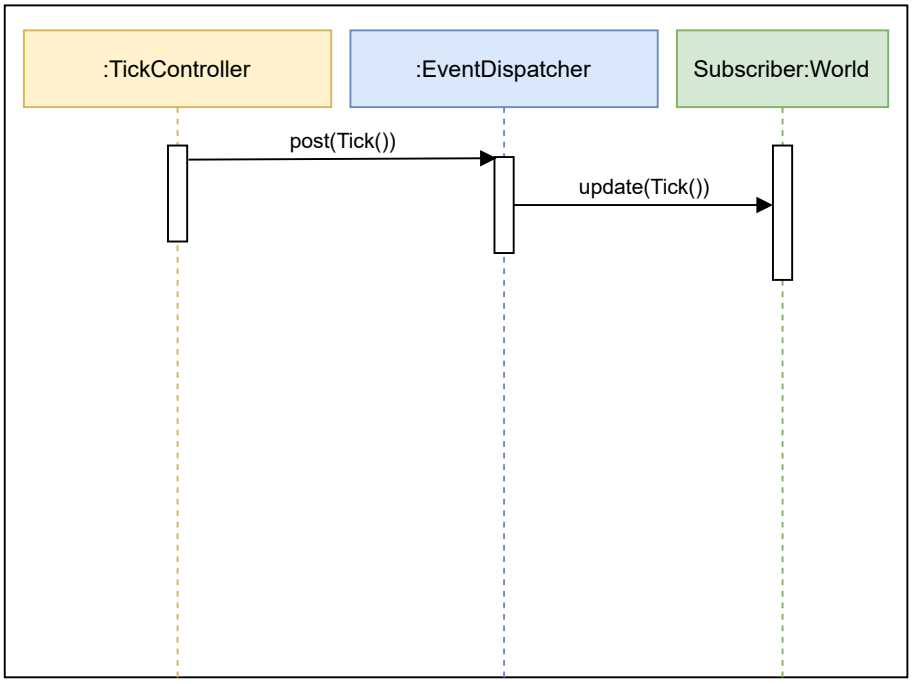
The EventDispatcher will send this event, only to the methods that have subscribed to it.

The two methods used in the EventDispatcher, Add, to subscribe an object to the event, and Post sends an instance of the event, to every suscriber of the event, to activate.



Subscription to an event.

Publish of an event.



Why use EventDispatcher

We use the EventDispatcher instead of the event system in pygame in order to make a more efficient code, since the subscriber only gets called to the events its interested in.

WeakBoundMethod

The WeakBoundMethod, holds only a weak reference of the object, so the eventDispatcher does not prevent it been deleted

Example

To subscribe an object to an event, you do this:

self.ed = event_dispatcher
self.ed.add(GameStart, self.game_start)

To send an instance of the event to the subscribers, you do this:

self.ed = event_dispatcher
self.ed.post(GameStart())

References: Python duck-typing for MVC event handling in pygame