 *copies A-E*

2-29-2000

TimedEvent Behavior 1.0 Event id's:N/A

By Matthew Peterson

matthew@pinoko.berkeley.edu

Introduction

This behavior provides a way of executing an event after an elapsed period of time. The time can be set on the fly. I made multiple copies of this behavior so that you can have multiple timers in the same movie. Copies A through E:

This behavior uses the following global variables to set and start the timer:

Global Variables:
1) ATimerTicks-- Set to number of ticks to elapse.
2) ATimerStart – Set to true to start the timer

Note: ATimerTicks will be called BTimerTicks in copy B of this behavior. The same with ATimerStart.

Purpose

Have you ever wished there was a way to wait a few seconds and then execute event? This is a common programming structure in multi-media. For instance, we might want our wired movies to time out and reset when not being used for a few minutes. Or after executing a gotourl command, we might want to check after several seconds if we actually went to that url or not. Timers have many obvious uses.

Quick Reference

Parameters:
1) `Atimer Alarm Event ID:` --- Set this parameter to the ID of the custom event you want executed as the timers "alarm".

Global Variables:
1) ATimerTicks -- Set to number of ticks to elapse before the alarm event is executed. Ticks are 1/60th of a second, so setting this variable to 60 means the alarm will be executed after 1 second. This variable is reset to zero after the timer has fired its alarm. It is therefore necessary to reset the

timer each time before you start it. You can check to see if the time is still running by seeing if ATimerTicks is greater than zero or not:

```
IF(ATimerTicks)
    //timer is still running.
ENDIF
```

Once a timer is running, it is not recommended to change ATimerTicks except under one situation, if you want to cancel the timer. To cancel the timer, set Atimerticks to zero. This will stop the timer, reset the time, and the alarm will not be executed.

2) AtimerStart – Set this to true to start the timer. The timer will begin counting from the next idle event. Once the timer is started, this value will be reset to false. Setting AtimerStart to true while the timer is running will cause it to start over with the new time, and the alarm will not be executed.

NOTE: I didn't make a looping timer, but it is easy to do. Just reset the timer in the custom event.

Reserved Variable Names for this Behavior:

GlobalVariables: KeyBoardOn , WhichKeyN, oldKeyN, IsCap, Ascii

Note: Internal variables all start with "MP_" and thus should not overlap with other variables.

Technical Notes:

The accuracy of the timer is dependent on the idle rate of the sprite track, how much processor time quicktime is using, and the processor speed.

I only provided 5 copies of this behavior (A-E). More can be made by simply replacing every occurrence of ATimer in the copy A of this behavior with XTimer, where X is your new prefix letter.

Revision History:

2-29-2000 Version 1.0 first written.