# Bite By Byte

powered by Adventure Maker users

issue 11
2009

## In This Issue




## Editors

Lyberodoggy
Chromegloss55
Mercedes
Mystery
Jaked
Saturn

**Fun with Math & Physics**
**by Lyberodoggy**

## *Editors Message*

It's been quite a month!
First, I was awarded an editor's position. Thank you, Chickens!
Secondly, the forum community awarded me in 3 categories. Thank you, everyone!
Lastly, I was awarded 'Member of the Month' for July! Thank you, AM Forum Staff!

It seemed like it was my last push to be active before I was whisked away from my 'work-from-home' contract! But, I'm trying to get reorganized so I can still make my presence known here. (rolls eyes)

Since I started last winter, this forum has been an excellent resource thanks to all the brilliant members. There has been a lull at the forums the past few weeks, but it's probably because everyone is either taking a break, working hard (raises hand), or off working on a project (raises hand again).

As co-editor, I'll do what I can to keep the AM Magazine looking sharp. Please consider adding a small article of your own. Anything helpful, informative, or fun!

See you online!
~reneuend

## *Back to school*

*By Lyberodoggy*


Math and Physics. Two subjects loved and hated equally. But what about them inside the video gaming industry?

AM is a powerful engine, that can accomplish from very simple things using mouse clicks to extremely advanced things using a touch of scripting.
Most of the users will never use anything more complex than some simple arithmetic equations -not necessarily because of ignorance, but because of the simplicity of most adventure games.
Let's admit that full 3d games use Physics more often than 2d ones. But what about creating complex puzzles?

This series of articles will teach you how to integrate useful physics laws inside your code and give you ideas about new puzzles.
Let's start with the most common. Newton's law.
According to the famous man, the acceleration of a body equals the sum of all forces acting on it divided by its mass, or

```
a=SF/m
```

So, here comes Galilian's experiment. The actual weight of a body is the force of Earth in it, which means the gravity acceleration times mass

```
W=m*g
```

Now, how can you use those? Well, right now you can't do much, but we 'll be learning kinematics soon enough and we 'll need this basic knowledge.
So let me introduce the basic equations:

**Simple linear movement**

```
u=Dx/Dt
```

**Simply accelerating linear movement**

```
a=Du/Dt
Dx=1/2a*Dt^2
```

**Legend**

**D** - difference (`Dvalue=lastvalue-firstvalue`)
**a -** acceleration
**u -** velocity
**x -** position
**t -** time

**Measurement**
**t** in milliseconds
**x** in pixels (or twips*15)
**u** in pixels/millisecond
**a** in pixels/millisecond^2

**Example of code for a linear movement**
Declare integers u,t

```
u=5*15
Action.CreateTimedEvent 0.01,"t=t+10:hotspot(1).left=u*t/1000",True
```

This will move the hotspot by 5 pixels a second. Of course that's a pretty small velocity. I recommend 40p/s for smooth movement in a 640*480 project.

**Example of code for accelerating linear movement**
Declare integers a,t

```
a=5*15
Action.CreateTimedEvent
0.01,"t=t+10:hotspot(1).left=1/2*a*(t/1000)^2",True
```

This is going to move your hotspot with an acceleration of 5p/s^2

**Diagonal simple movement example**
This is our first complex movement. It consists of two linear movements. One vertical and one horizontal. This movement is based on the principle of movements' independence. The two movements won't affect each other
Declare integers ux,uy,t

```
ux=5*15
uy=5*15
Action.CreateTimedEvent
0.01,"t=t+10:hotspot(1).left=ux*t/1000:hotspot(1).top=uy*t/1000",True
```

this will relocate the hotspot by adding 5 pixels per second to both its coordinates.
Due to the principle we described though, you can alter ux and uy independently. For instance they could be set to 25 and 60 p/s accordingly
Also, you can have hybrid movements, by using simple linear horizontal and linear accelerating vertical movements or vice versa.
Example:
Declare integers ux,ay,t

```
ux=20*15
ay=10*15
```

```
Action.CreateTimedEvent
0.01,"t=t+10:hotspot(1).left=ux*t/1000:hotspot(1).top=1/2*ay*(t/1000)^2
",True
```

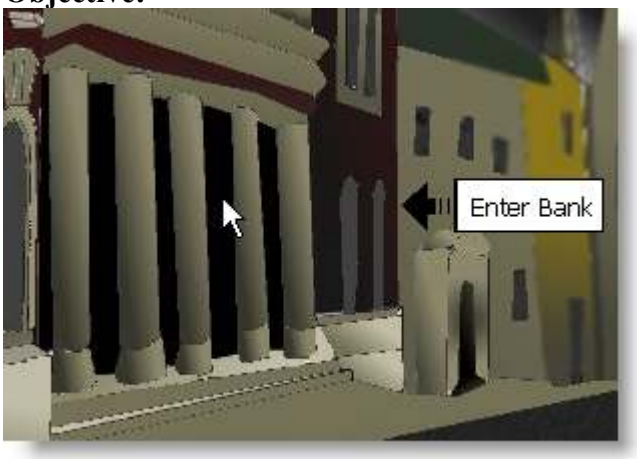
In the next lesson, we 'll show how to affect accelerating moves by adding or removing forces and the gravity acceleration.

## KeyGuard Step by Step (Part 2): Mouse Hover

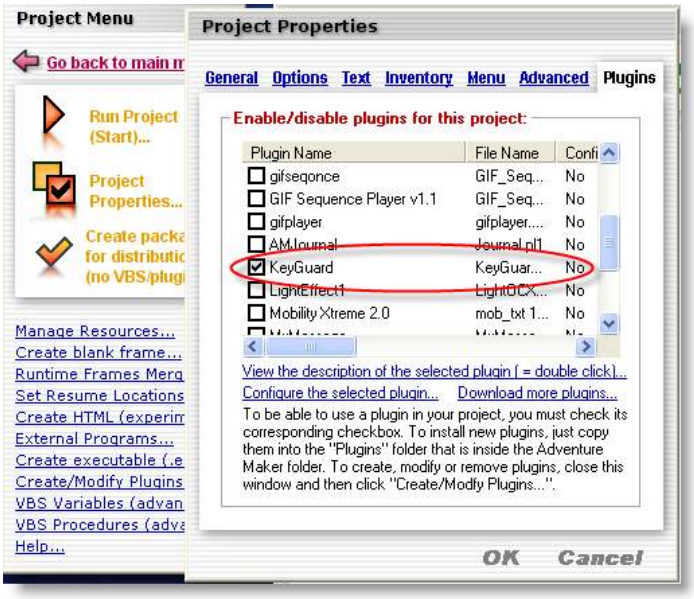
By reneuend

In this tutorial, we will look at a useful application of the “MOUSE HOVER” from ShadowHunter’s **KeyGuard** plugin.

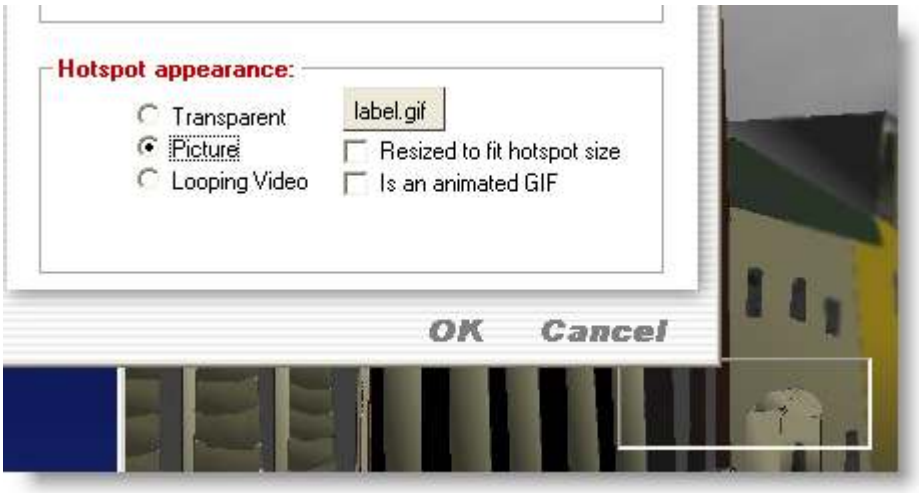
**Objective:**



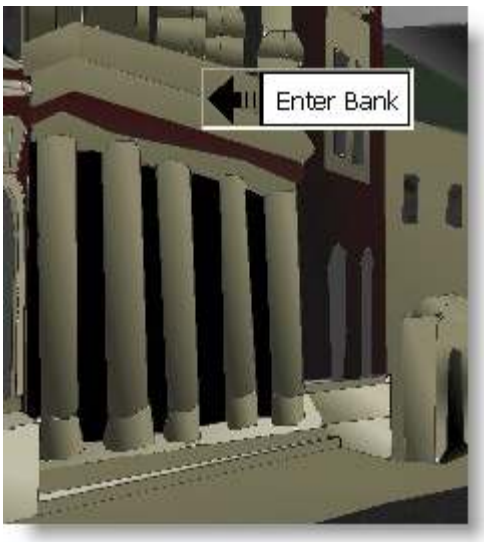

When the player moves the mouse over a doorway, a label will display showing the player  the name of the building and an arrow to signify that they can click to enter!



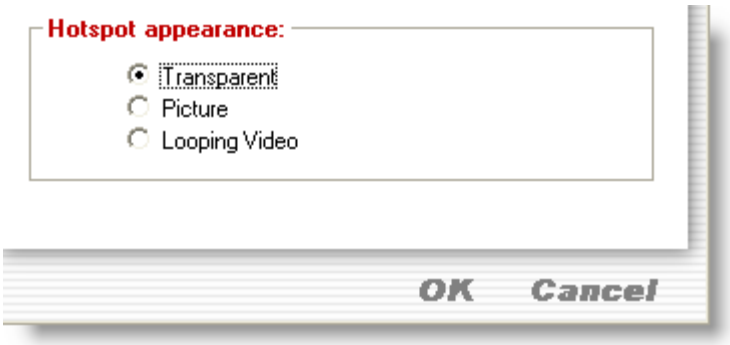

Select the KeyGuard Plugin for your project!




Add a label to the hotspot that you want displayed when the mouse hovers over the doorway. Please keep note of the hotspot index number!

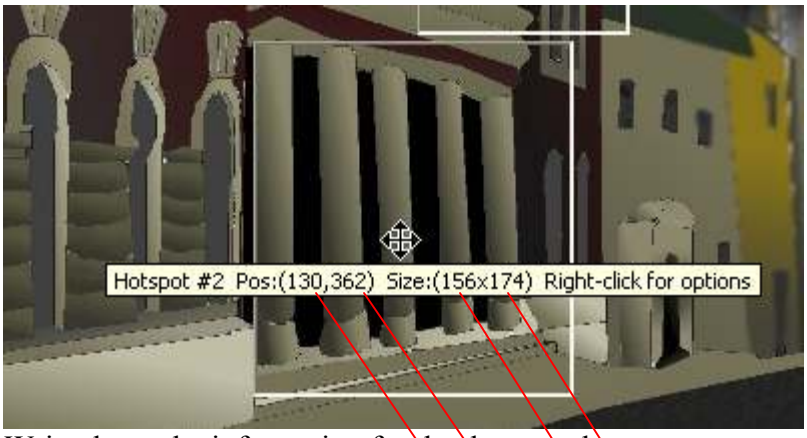



Position the label where you like it.




Now make the hotspot transparent because it will only display the label image when the mouse hovers over it.


Make another hotspot for the doorway. When the player hovers over this hotspot, the "label" hotspot will display the "Enter Bank" image.




Write down the information for the doorway hotspot so we can set it up with KeyGuard.

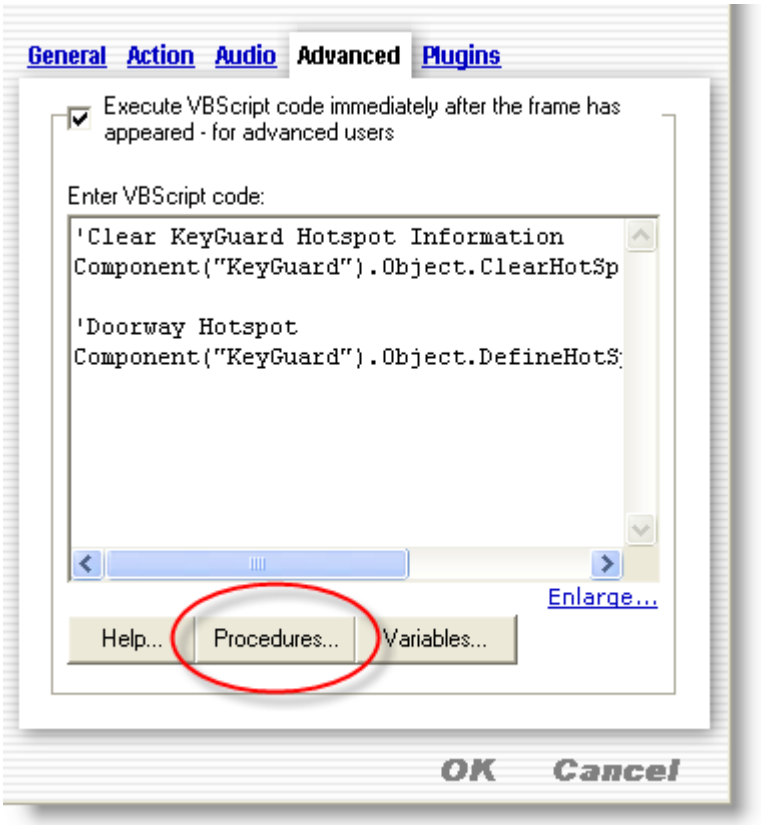
**VBScript Code**

```
'Clear KeyGuard Hotspot Information
Component("KeyGuard").Object.ClearHotSpots

'Doorway Hotspot
Component("KeyGuard").Object.DefineHotSpot 130,362,156,174,2,0
```

Hotspot Number  Non-Recurring

Use the information you wrote down about doorway hotspot to enter the KeyGuard hotspot definition in the Frame Properties – Advanced Tab.

Note: Non-Recurring means > 0=hover action occurs everytime hotspot is hovered
1=hover only occurs one time the hotspot is hovered




Now we will enter the “hover” event code. Click on the “Procedures” button.

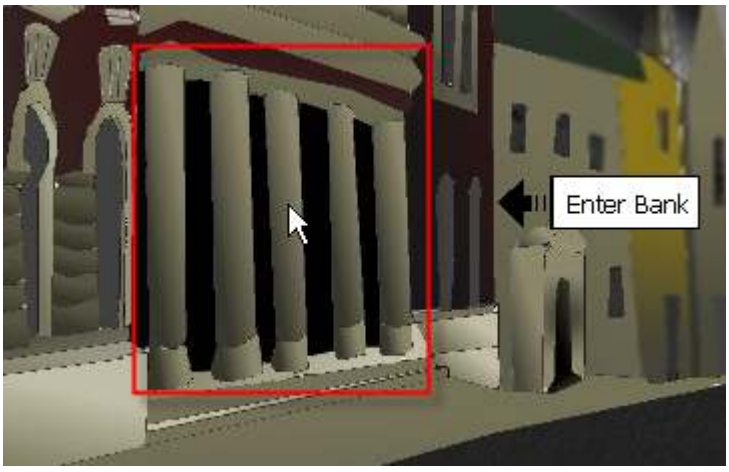
```
Sub KeyGuard_MouseEntersHotSpot(HotspotTAG)
	'MsgBox "Mouse enters hotspot area: " & HotspotTAG(0)

	If Action.GetCurrentFrameName = "City Street" Then
		If HotSpotTag(0) = 2 Then Action.LoadAPicture Hotspot(1), "label.gif"
	End If


End Sub
```

Enter the “IF” condition to check for the frame name
Then enter another “IF” condition inside this one to see if the mouse is hovering over the doorway hotspot. If it is hovering over the doorway hotspot, then display the label image in the “label” hotspot.

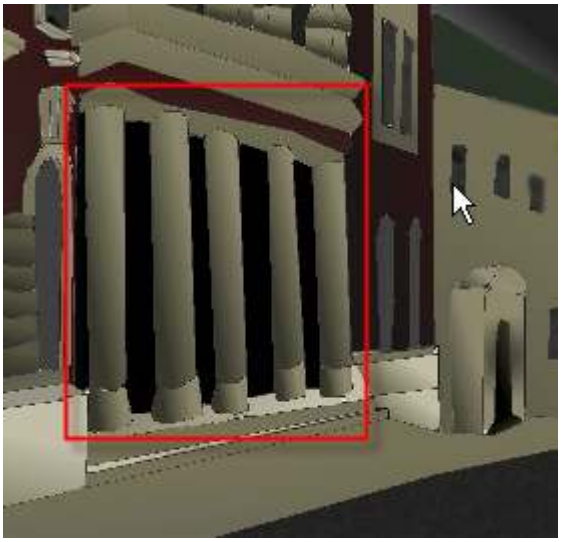
```
Sub KeyGuard_MouseLeavesHotSpot(HotspotTAG)
	'MsgBox "Mouse leaves hotspot area: " & HotspotTAG(0)

	If Action.GetCurrentFrameName = "City Street" Then
		Action.LoadAPicture Hotspot(1), ""
	End If
End Sub
```

If the mouse moves outside the doorway hotspot, we don’t want to continue displaying the label! So, We check for the framename again with the “IF” condition and then we clear out the hotspot that has the label image.

**TEST:**




When you move the mouse within the doorway hotspot, the label displays!



If the mouse is outside the doorway hotspot area, the label no longer displays!

**Conclusion:**

There are many variations for using the mouse hover from KeyGuard. You could for instance use a compass and change the compass arrow depending on where the mouse is hovering on the screen.

Next time, we will look at KeyGuard's Image Cursor! For an example, you can download the game I created with it called, **Texas Shoot 'Em** at http://www.reneuend.info.