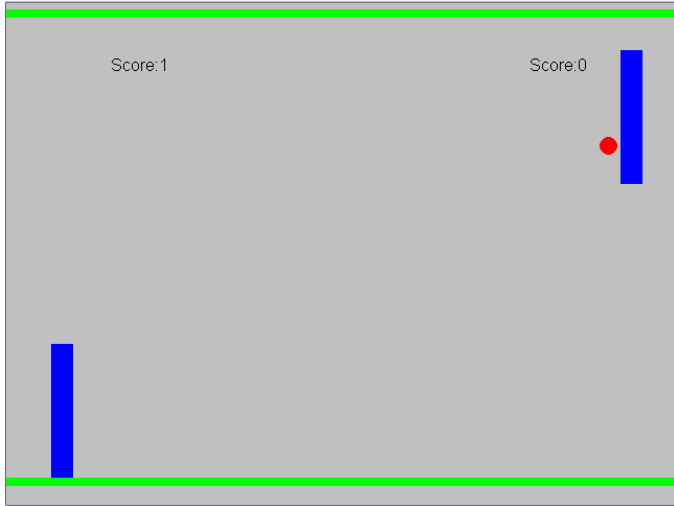


# Game Maker Tutorial

## Pong

Fall 2007 IAT410  
Week 4 Lab

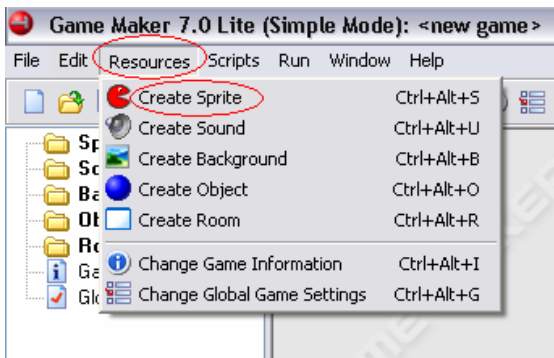


## Sprites

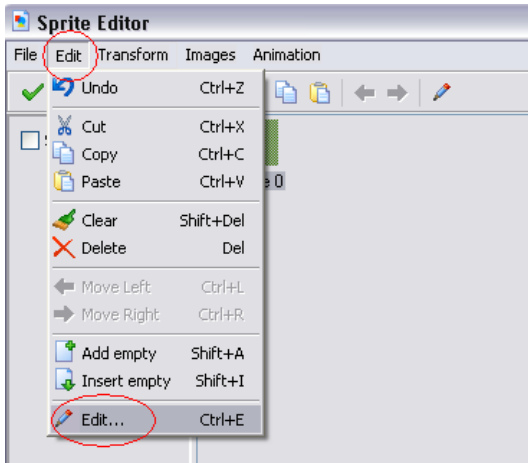
Sprites are like little images. You can either import or create/edit these images. We'll create three images: ball, line, and paddle.

### Creating the ball sprite resource for the game:

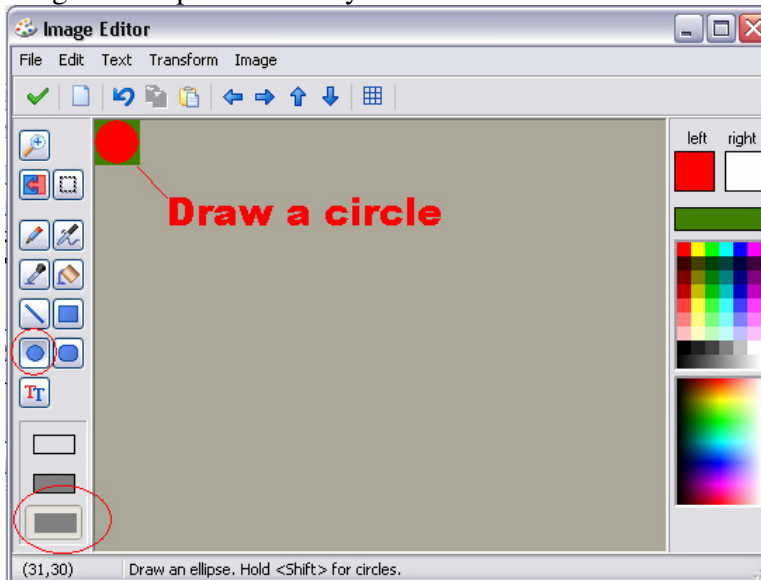
1. From the **Resources** menu, choose **Create Sprite**. The Sprite Properties form appears.





2. Click on the **Name** field where currently says `sprite0`. This is the default name for the sprite. Rename it to `spr_ball`.
3. Click on the **Edit Sprite** button. This opens the Sprite Editor.
4. Click on **Edit** and choose **Edit....** ( or double-click on the image)



5. Use the ellipse tool to draw a circle. Do not worry about the background color, since the image is transparent unless you click on the checkmark next to **Transparent** to remove it.



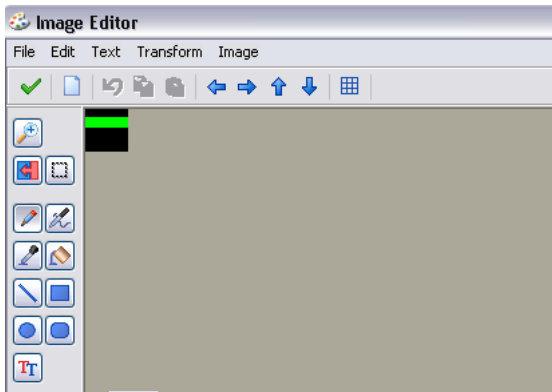
6. Press  at the left top to close the image editor.



7. Press  at the left top to close the sprite editor.

### Creating the line sprite for the game:

Multiple line sprites will together form a line.

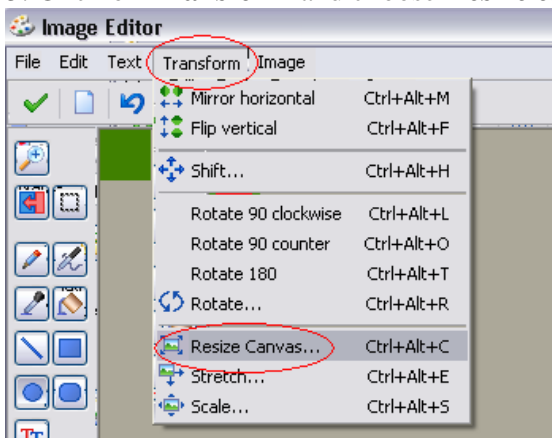
1. From the **Resources** menu, choose **Create Sprite**. The Sprite Properties form appears.
2. Click on the **Name** field and rename it to `spr_line`.
3. Click on the **Edit Sprite** button. This opens the Sprite Editor.
4. Click on **Edit** and choose **Edit....**
5. Use the rectangle tool to draw a line (thin rectangle).





6. Press  to save and close the image editor.
7. Press  at the left top to close the sprite editor.

### Creating the paddle sprite for the game:

1. From the **Resources** menu, choose **Create Sprite**. The Sprite Properties form appears.
2. Click on the **Name** field and rename it to `spr_paddle`.
3. Click on the **Edit Sprite** button. This opens the Sprite Editor.
4. Click on **Edit** and choose **Edit...**
5. Click on **Transform** and choose **Resize canvas**. Resize to 400% and click OK.



6. Use the rectangle tool to draw a paddle.
7. Press  at the left top to close the image editor.
8. Press  at the left top to close the sprite editor.

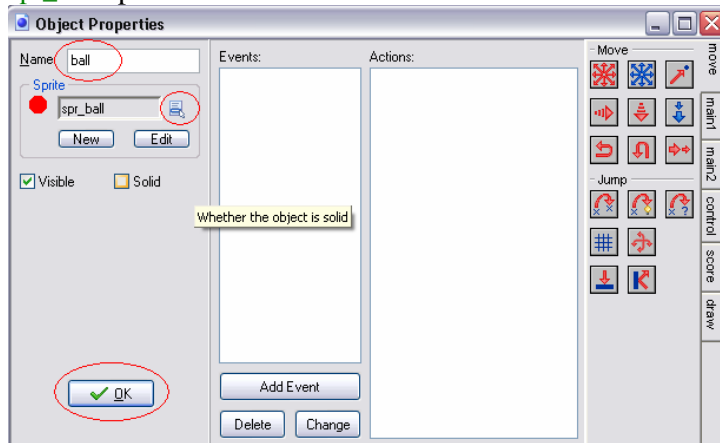
Go to **File** and choose **Save as...** to save your work.

Having created the sprites does not mean that anything is happening. Sprites are only the images for game objects and we have not yet defined any game objects.

## Objects

### Create the ball object:

1. From the **Resources** menu, choose **Create object**. The Object Properties form appears.
2. Click on the **Name** field and rename the object to **ball**.
3. Click on the icon at the end of the **Sprite** field and in the list of available sprites, select the **spr\_ball** sprite. Press **OK** to close the form.



### Create the paddle objects:

1. From the **Resources** menu, choose **Create object**. The Object Properties form appears.
2. Click on the **Name** field and rename the object to **paddleR**.
3. Click on the icon at the end of the **Sprite** field and in the list of available sprites select the **spr\_paddle** sprite.
4. Press **OK** to close the form.
5. Create **paddleL** object like you created the **paddleR** object .

### Create the line object:

1. From the **Resources** menu, choose **Create object**. The Object Properties form appears.
2. Click on the **Name** field and rename the object to **line**.
3. Click on the icon at the end of the **Sprite** field and in the list of available sprites select the **spr\_line** sprite.
4. Instances of the line object must be solid, that is, no other instances should be allowed to penetrate them. To this end click on the box next to the **Solid** property to enable it.
5. Press **OK** to close the form.

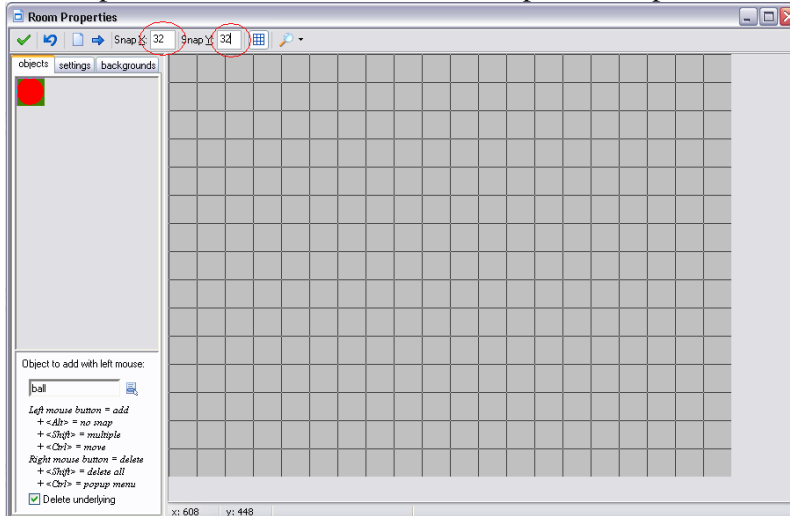
## Room

Now that we have created the game objects there is one more thing to do. We need to create the room in which the game takes place.

### Creating the room:

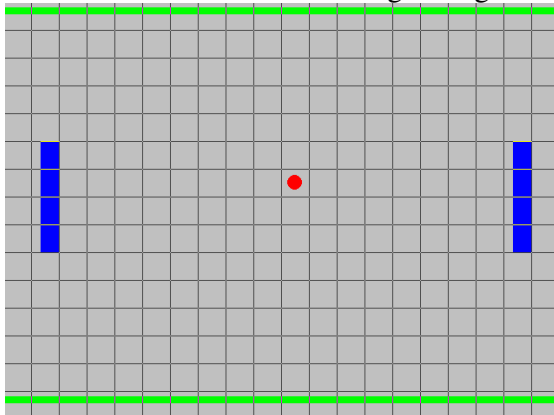
1. From the **Resources** menu choose **Create Room**. The Room Properties form will show.
2. On the left you see three tabbed pages. Select the page labeled **settings**. In the **Name** field type in **rm\_main**. In the **Caption for the room** field type 'Pong'.


3. Select the **objects** tab. Enlarge the window somewhat such that you can see the complete room area at the right. At the top, change the value for **Snap X** and **Snap Y** to 32. As the size of our sprites is 32, this makes it easier to place the sprites at the correct locations.



4. At the left you see the image of the ball object. This is the currently selected object. Place one instance of it in the room by clicking with the mouse somewhere in the centre of the grey area.

5. Click on the icon with the menu symbol next to the field **ball**. Here you can select which object to add. Select **line**. Click on the different cells bordering the room to put instances there. To speed this up, press and hold the <Shift> key on the keyboard and drag the mouse with the mouse button pressed. Also place **paddleR** and **paddleL**. You can remove instances using the right mouse button.



6. Press  at the left top to close the form.

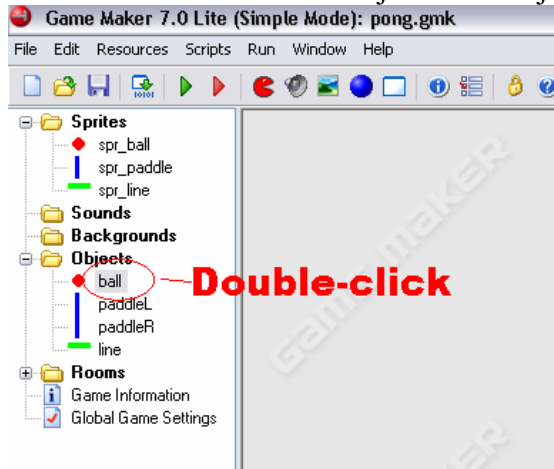
## Actions

Instances of game objects don't do anything unless you tell them how to act. You do this by indicating how the instances of the object must react to *events* that happen. There are many different events that can happen. The first important event is when the instance is created. This is the **Create Event**. Probably some action is required here. For example we must tell the instance of the ball object that it should start moving in a particular direction. Another important event happens when two instances collide with each other; a so-called **Collision**

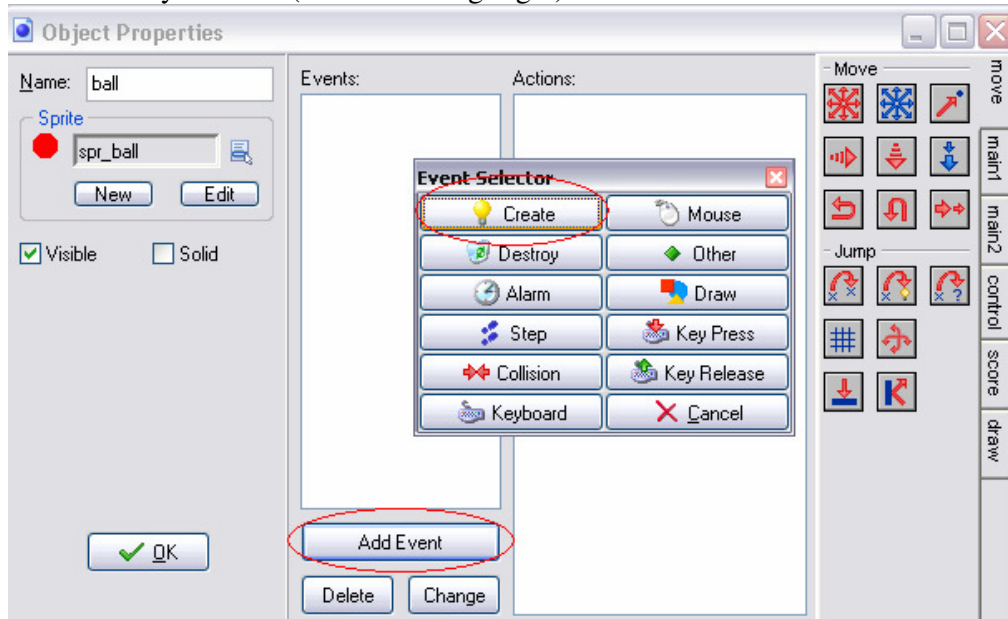
**Event.** For example, when the instance of the ball collides with an instance of the paddle, the ball must react and change its direction of motion.


### Let the ball object move:

1. Double-click on the **ball** object under objects. Press the **Add Event** button.

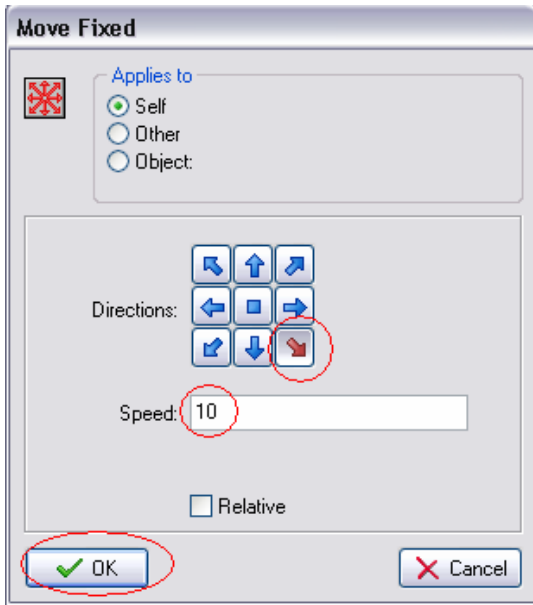



2. Click on the **Create** button. The create event is now added to the list of events. It is automatically selected (with a blue highlight).



3. Next you need to include a **Move Fixed** action  in the list of actions. To this end, press and hold the mouse on the action image with the eight red arrows in the page at the right, drag it to the empty actions list, and release the mouse. An action form is shown asking for information about the action.

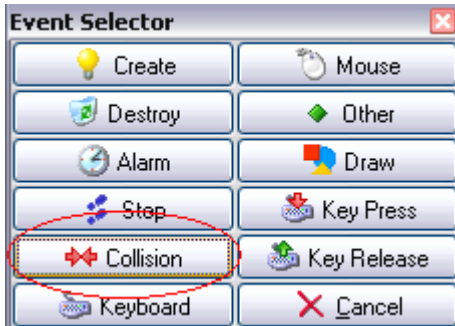
4. In the action form for the **Move Fixed** action you can indicate in which direction the instance should start moving. Select the lower right one. Note that the selected directions turn red. When multiple directions are selected one is chosen randomly. Also set the **Speed** to 10. Press **OK** to indicate that we are ready with this action.




Next we need to test the game. Testing is crucial. Testing (or running the game in general) is simple; choose the command **Run normally** from the **Run** menu (or use the shortcut icon ). The design window will disappear, the game will be loaded and, if you did not make any mistakes, the room will appear on the screen with the ball moving inside it. Test the game regularly from now on.



### Handling a collision with the line:

1. Press the **Add Event** button. In the Event Selector click on the **Collision** button and select **line**. The collision event is now added to the list of events.




2. Include a **Bounce** action  by dragging it from the page at the right. The action form will appear. There are two properties we can change but their default values are fine. We are not interested in precise bounces and we want to bounce against solid objects. (Remember that we made the line object solid.) Press **OK** to close the action form.


### Handling a collision with the paddles:

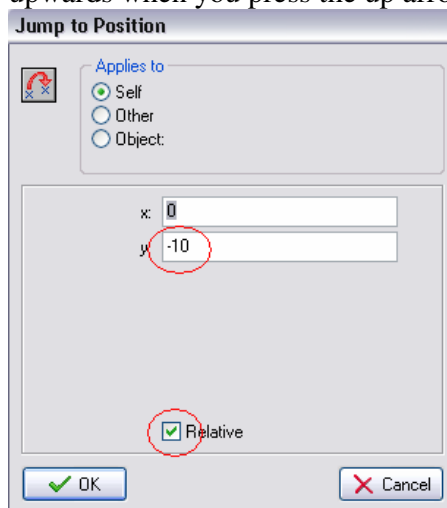
1. Press the **Add Event** button. In the Event Selector click on the **Collision** button and select **paddleR**. The collision event is now added to the list of events.
2. Include the **Bounce** action  by dragging it from the page at the right. The action form will appear. You must change the property **against** to **all objects**. We are not interested in precise bounces, but we want to bounce against a non-solid object. Press **OK** to close the action form.
3. Press the **Add Event** button. In the Event Selector click on the **Collision** button and select **paddleL**. The collision event is now added to the list of events.
4. Include a **Bounce** action  by dragging it from the page at the right. The action form will appear. You must change the property **against** to **all objects**. Press **OK** to close the action form.

### Restarting the game when the ball goes out the screen:

1. Press the **Add Event** button. In the Event Selector click on the **Other** button and select **Outside Room**.
2. Include a **Jump to Start** action  by dragging it from the page at the right. Press **OK** to close the action form. Press **OK** to close the Object Properties panel.

### Moving paddleL by pressing up and down arrow keys

1. Double-click on the paddleL object. Press the **Add Event** button. In the Event Selector click on the **Keyboard** button and select **<Up>**.
2. Include the **Jump to Position** action  by dragging it from the page at the right. This opens up the Jump to Position action form.
3. Set **y** to -10 (remember the upper-left corner is at (0, 0)?) and click on the box next to the property **Relative** to enable it. Press **OK** to close the form. Now the paddle should move upwards when you press the up arrow key.




The screenshot shows a dialog box titled "Jump to Position". It has a small icon in the top-left corner. Below the icon is a section labeled "Applies to" with three radio buttons: "Self" (selected), "Other", and "Object:". Below this are two input fields: "x:" with the value "0" and "y:" with the value "-10". At the bottom of the dialog, there is a checked checkbox labeled "Relative". At the very bottom are two buttons: "OK" and "Cancel".




4. Follow similar steps to make the paddle move downwards when the down arrow key is pressed.  
Press **OK** to close the object properties panel.

### **Prohibiting the paddleL from going out of the screen.**

1. Press the **Add Event** button. In the Event Selector click on the **Collision** button and select line.
2. You need to include the **Move Fixed** action  in the list of actions.
3. In the action form for the **Move Fixed**, Select the middle one. Also set the **Speed** to 0. Note that this will stop the paddle. Press **OK** to indicate that we are ready with this action. Press **OK** to close the Object Properties panel.


### **Making the paddleR follow the ball movement**

1. Double-click on the paddleR object. Press the **Add Event** button. In the Event Selector, click on the **Step** button.
2. Include the **Execute code** action  under **control** tab by dragging it from the page at the right. This opens up the editor. Type the following piece of code:

```
{  
    move_towards_point(paddleR.x, ball.y, 5);  
}
```

`move_towards_point(x, y, sp)` moves the instances with speed `sp` toward position `(x,y)`. Now the paddleR follows ball movements (because it follows itself in terms of `x`, it should only move up and down). `5` indicates the speed. When the paddleR moves faster, it is unlikely to miss the ball. If you want the paddleR's center to follow the ball instead of its top, rewrite the code as follows:


```
{  
    move_towards_point(paddleR.x, ball.y - (paddleR.sprite_height/2.0), 5);  
}
```

Press  to save and close the editor.

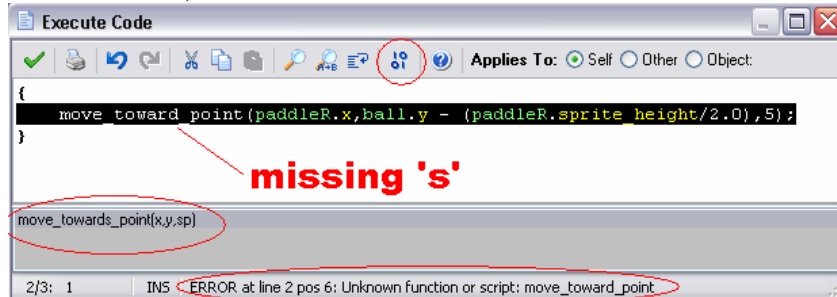
### **Prohibiting the paddleR from going out of the screen.**

Please refer to **Prohibiting the paddleL from going out of the screen** described above.

## Debugging

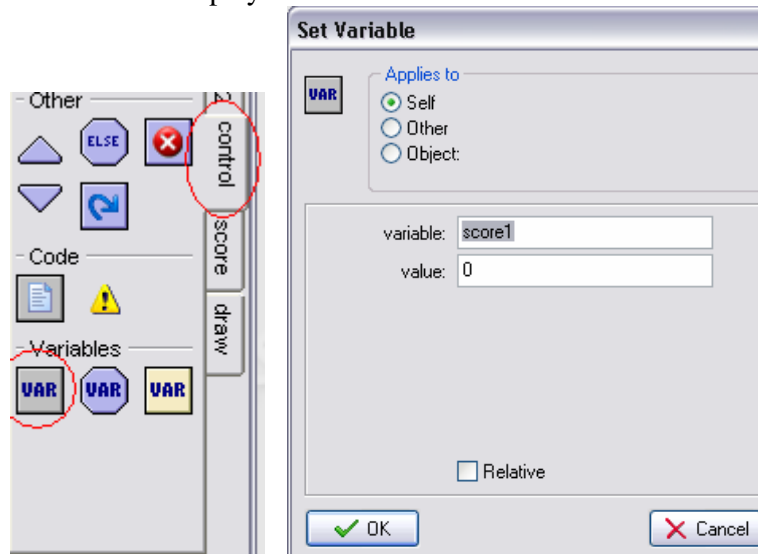
While writing code, you can always press **Check the script for syntax errors** button  to see if there is any bugs in your code. It usually tells what the bug is and where it is located.


Example error: Unknown function `move_toward_point` (`move_towards_point` is the correct function name)



## Counting scores

1. Double-click on the ball object.
2. Choose **Create event**
3. Include **Set Variable** action under control by dragging it from the page at the right. In the action form for the **Set Variable**, set the **variable** to `score1`. Also set the **value** to 0. Press **OK**. `score1` is the score for the player.



4. Include one more **Set Variable** action. In the action form for the **Set Variable**, set the **variable** to `score2`. `score2` is the score for the computer. Also set the **value** to 0. Press **OK**.
5. Click on **Add Event** and add **Step** event. The **Step** event happens every step of the game. Here you can put actions that need to be executed continuously. Include the **Execute code** action  under **control** by dragging it from the page at the right. This opens up the editor. Type the following piece of code:

```
{  
    draw_set_font(-1);  
}
```


```


draw_set_color(c_black);
draw_set_alpha(1.0);
scoreStr = string_insert(string(score1), 'Score: ', 7);
draw_text(100, 50, scoreStr);
scoreStr = string_insert(string(score2), 'Score: ', 7);
draw_text(500, 50, scoreStr);
screen_refresh();
}

```

**draw\_set\_font(font)** sets the font that will be used when drawing text. Use -1 to set the default font (Arial 12).

**string\_insert(substr, str, index)** returns a copy of str with substr added at position index. **draw\_text(x, y, string)** draws the string at position (x,y), using the drawing color and alpha. Note that the upper left corner of the screen is at position (0, 0). Now the scores can be shown on the stage, but they are both 0.

6. Press  to save and close the editor. Now you'll need to increment the scores.

7. Choose **Outside Room** event. Include the **Execute code** action  under **control** and type:

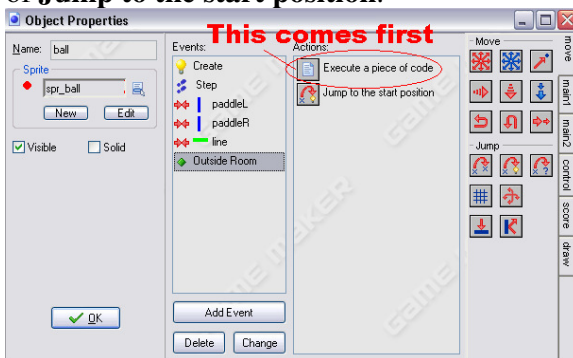
```

{
    if(ball.x < 0) {
        score2 = score2 + 1;
    }
    else {
        score1 = score1 + 1;
    }
}

```

The code above will increment the *score2* by 1 if the ball goes beyond the left side of the screen, otherwise it will add 1 to the *score1*. Since the **Step** event we added previously happens every step of the game, we do not have to worry about redrawing the scores.

**Important:** The code must be executed before **Jump to the start position** action (otherwise it will always be caught by else {}); therefore, drag **Execute a piece of code** action to the top of **Jump to the start position**.



## Reference

Overmars, M. 2007. Game Maker Tutorial Your First Game. YoYo Games Ltd.