GameMaker Objects and Paths

Objects and Instances in GameMaker

In GameMaker (and most other game engines) one of the game elements are called Objects. An Object is an element in the game that consists of properties and methods and that can respond to events. In GameMaker the term Object references a pattern and not a particular object. This pattern consists of directions for the GameMaker game engine to create Instances of an Object.

When you start a new GameMaker game the general steps are:

1. Gather the game assets including sprites and backgrounds, music and sound.
2. Load these assets into the GameMaker editor.
3. Create a Room and start defining objects.
4. Set the Room with a background and drag objects into the room

Part of this process is defining objects. When defining an object the game designer will:

1. List the events the object will respond to
2. Associate an action with each event

When an object has been created and dragged into the room, GameMaker creates an instance of the object and places it into the room. The actual Object pattern cannot be placed in the Room. This means that when the object is dragged into the room multiple instances of the object are inserted into the room.

This and later examples will use images from the Braid Graphics zip file. This file contains images for the game Braid located at http://www.davidhellman.net/braidbrief.htm. Download the zip file or go to the web site and download graphics. These will include animated images, static images, backgrounds, and sprite sheets.

Note: this lecture will use the GameMaker version 8.1 and will include notes on GameMaker studio 2.

Get the assets

1. Download the braid graphics file (http://staffwww.fullcoll.edu/dcraig/gameprog/braidgraphics.zip) and unzip.
2. In the unzipped folder locate the file named **bg_4_02.jpg** and copy it to the desktop.
3. In the icons128 file locate the file named **door1.png** and the file named **flora3.png** and copy these to the desktop.
4. In the animation-set folder locate the file named **monster-walk-b.gif** and copy it to the desktop.

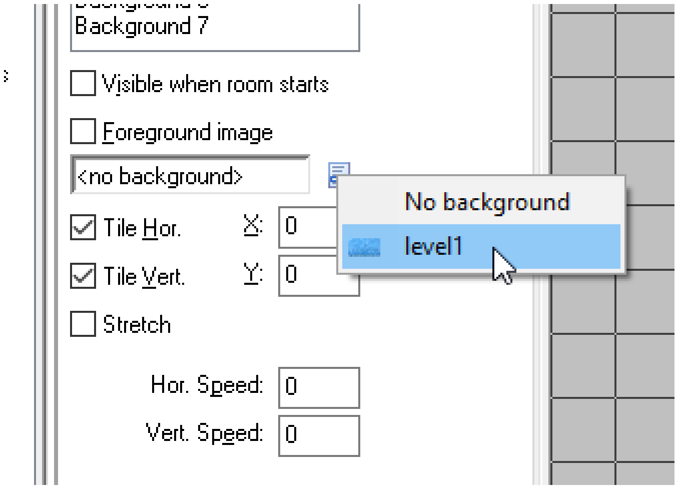
   You now have a background image, two static images (the door and flora) and an animated GIF image

Import the assets to GameMaker (8.1)

1. Start GameMaker.
2. Right click the Sprites folder and choose Create Sprite.
3. Name the sprite Door and click the Load Sprite button.
4. Choose the door1.png file and click the Remove Background check box. This removes the background for the sprite. Click Open.
5. In the sprite screen click the Open button to center the sprite origin point.
6. Follow these steps for the flora3.png file and name the sprite Flora.
7. Follow these steps for the monster-walk-b.gif file and name the sprite Monster.
8. Right click the Backgrounds folder and choose Create Background.
9. Name the background level1 and click the load background button.
10. Choose the background image bg_4_02.jpg and click Open. Note the size of the background image.

Creating a Room and a Background (8.1)

1. Right click the Room folder and choose Create Room and name it level1room
2. In the room menu click the Settings tab.
3. Set the room width to 1280 and the height to 720
4. Click the backgrounds tab.
5. Click the drop-down and choose Level1.



Background 7

□ Visible when room starts

□ Foreground image

<no background>

☑ Tile Hor. X: 0

☑ Tile Vert. Y: 0

□ Stretch

Hor. Speed: 0

Vert. Speed: 0

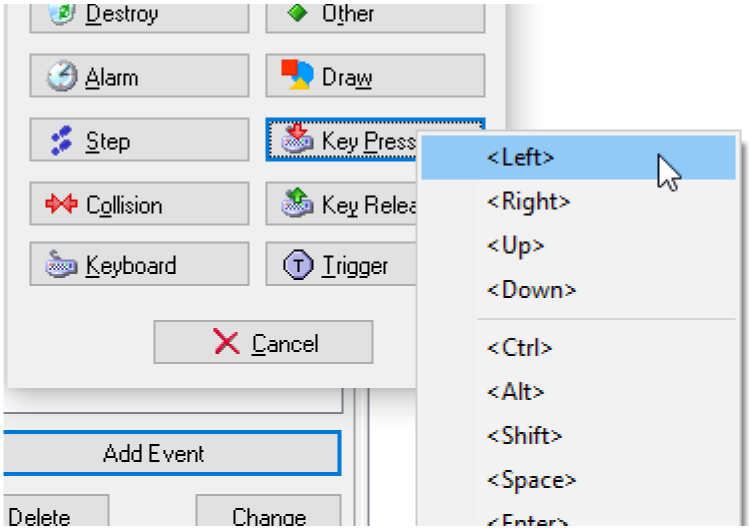
6. Click the Run button to run the game. Only the background will show since we have not yet created and added objects.

Creating and Adding Objects (8.1)

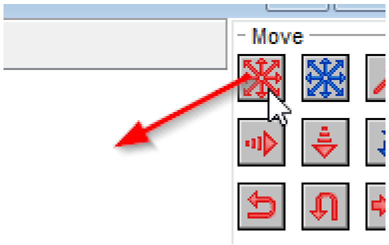
1. Right click on the Objects folder and choose Create Object. Name this object Doorobj.
2. Click the drop-down next to <no sprite> and choose the Door sprite.
3. Click OK to save this object.
4. Using the same steps create an object named Floraobj using the flora sprite and an object named Monsterobj using the monster sprite.

Programming Objects with the GUI (8.1)

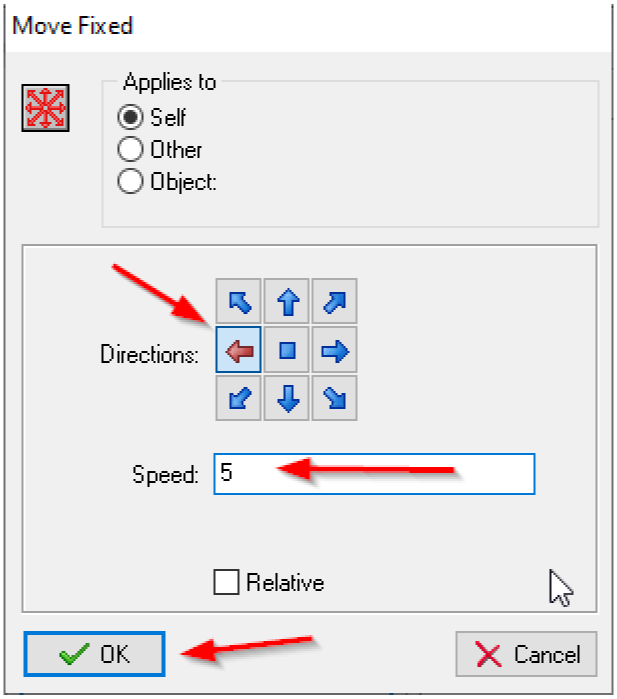
1. Double click the Monsterobj object to open the editor window.
2. Click the KeyPress event button and choose Left



3. Drag the Move button into the Actions area. A dialog box is displayed.



4. In the dialog box click the Left direction arrow, enter the speed at 5 pixels a second, and click the OK button.



Move Fixed

Applies to
● Self
○ Other
○ Object:

Directions:

Speed: 5

□ Relative

This causes the object to move to the left at 5 pixels a second when the left arrow key is pressed.

5. Click the button for Key Release in the Event section and select Left. This event is generated when the player released the left arrow key.

6. Drag the Move button to the Actions area. In the dialog box click the left arrow in the directions area but set the speed to 0. This stops the object from moving when the left arrow button is released.

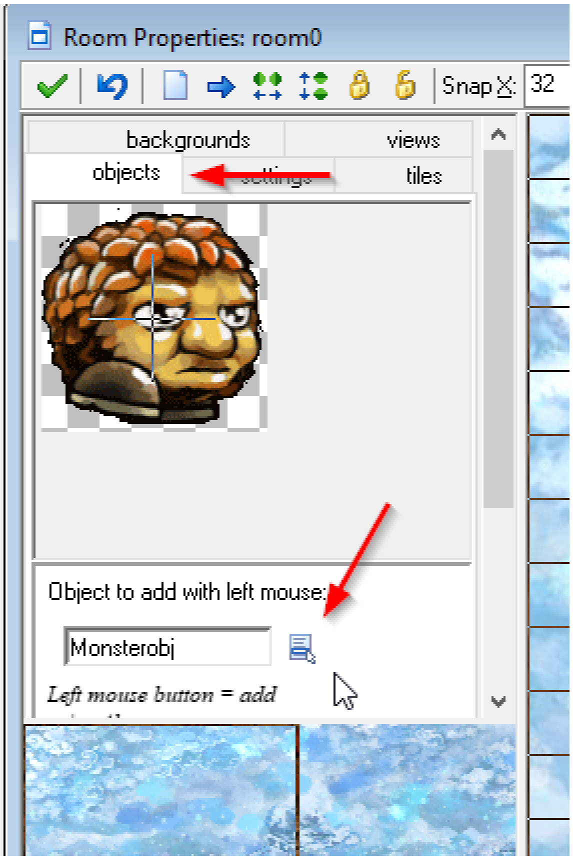
7. Click OK for the object properties.

8. Using these two steps add the Key Down and Key Release for the right arrow key for the Monster object.

Adding the object to the Room (8.1)

1. Double click the room object to open the room editor.

2. Click the Object tab and choose an object with the drop down.



Object to add with left mouse: Monsterobj

Left mouse button = add

3. Choose an object and then click in the room area to add the object to the room.

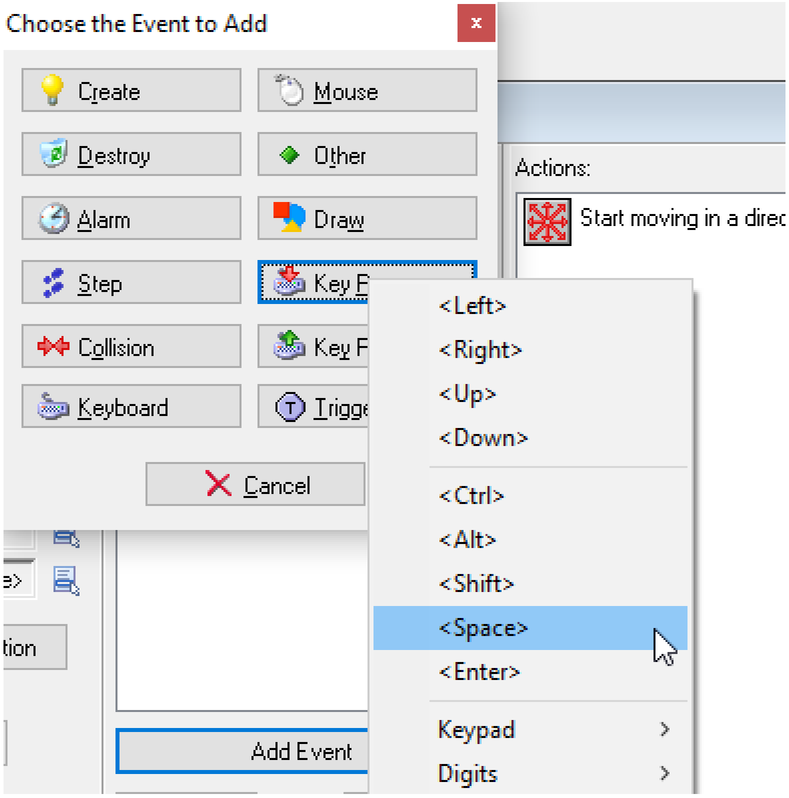
4. Add the Monster Object and the door object.

5. Run the game. You should be able to move the Monster left/right with the arrow keys.

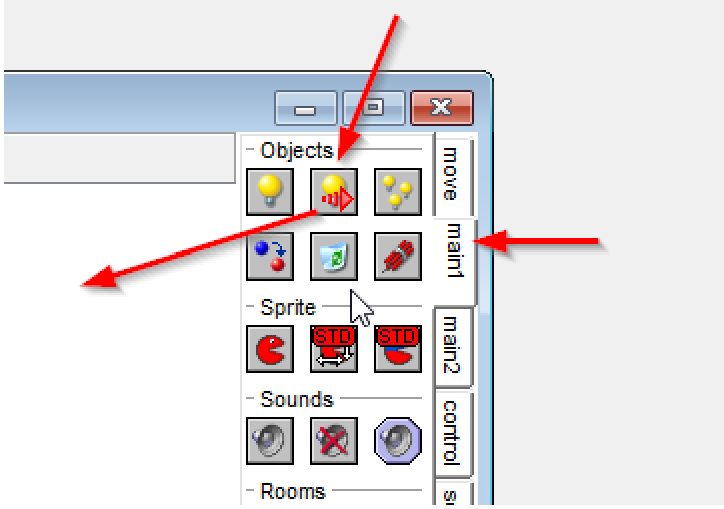
Creating a Moving Object (8.1)

1. Double click the Monster object to open the editor.

2. Click Add Event, choose KeyPress, and choose Space



3. Click the Main1 tab and drag the Creating Moving icon into the Actions area:

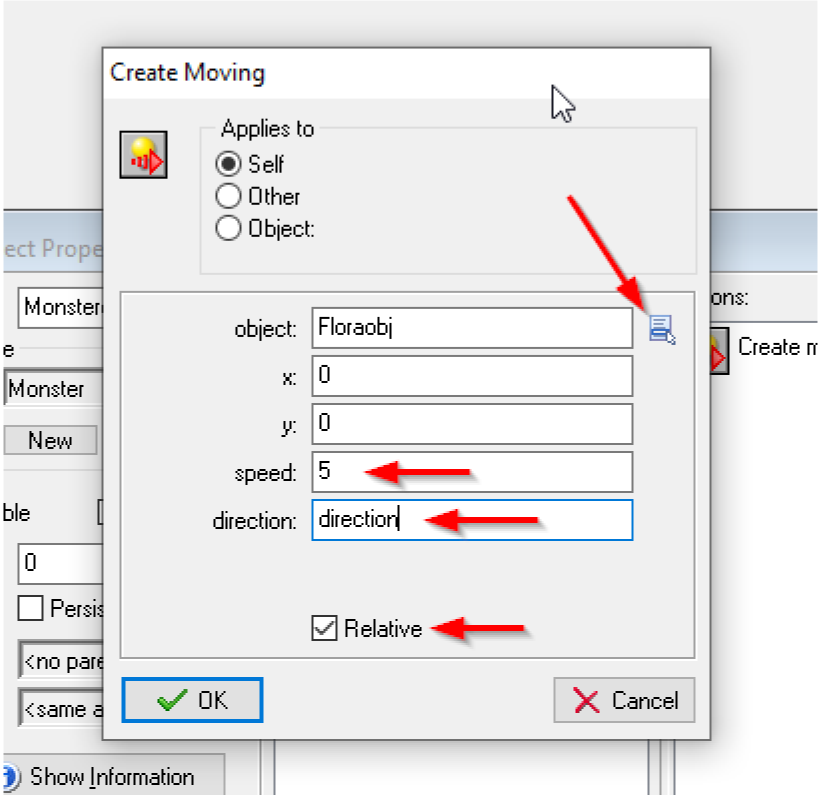


The create moving dialog box is used to create a moving object. The dialog box is used to set the position of the created object, the speed it will be moving, and the direction it will be moving.

4. Click the object drop-down to choose the Flora object. This will be object that will be created.

5. Click the relative check box to create the object at the position of the Monster object. You could have typed (in the x and y boxes) Monsterobj.x and Monsterobj.y.

6. Set the speed to 5 and type the word **direction** in the direction text box. This term is a property of Monsterobj that indicates which direction the Monster is moving. The value of direction is a degree value with 0 pointing to the right and 180 pointing to the left.



Create Moving

Applies to

● Self
○ Other
○ Object:

object: Floraobj
x: 0
y: 0
speed: 5
direction: direction

☑ Relative

7. Click OK to save the dialog box.

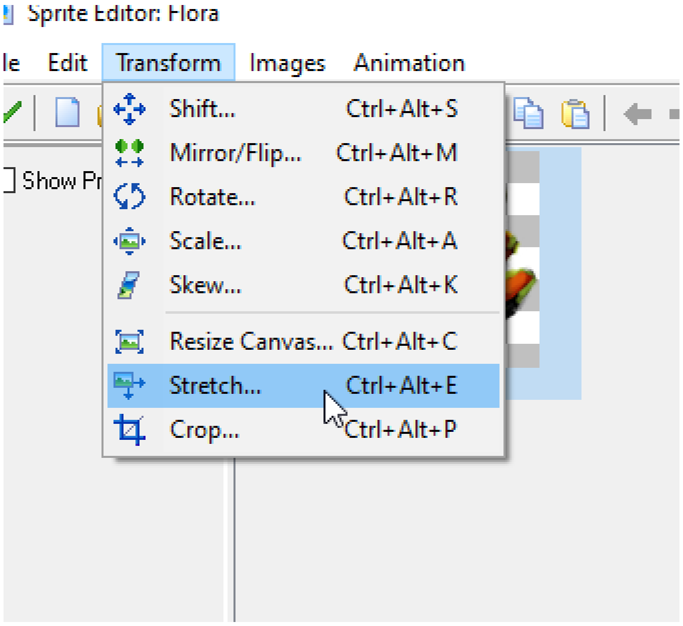
8. Run your game. When you move your monster to the left or right and press the spacebar an instance of the Flora object will be created with the specified speed and direction.

This little game looks ok but there are several things that still need to be done. These include mirroring the Monster so it faces the right direction and making the Flora sprite a bit smaller.

Reducing the size of the Sprite (8.1)

1. Double click the Flor sprite and click the Edit Sprite button.

2. In the Edit Sprite window click the Transform menu item and select the Stretch menu item.



3. For the width enter 30 to set the size of the sprite to 30% and click the green check mark.

4. In the sprite properties dialog box click the Center button to set the sprite origin.

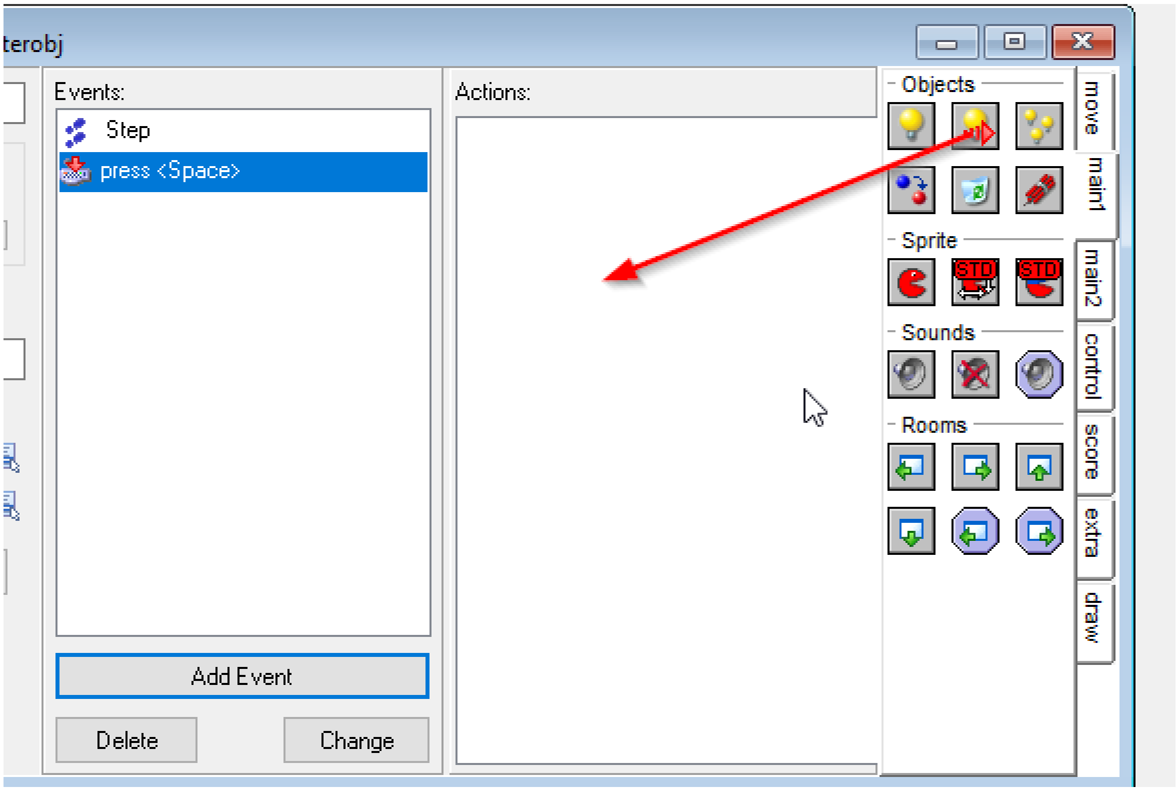
5. Run the game again and notice that the flora object is smaller.

6. Save this game with the name GameGUI.

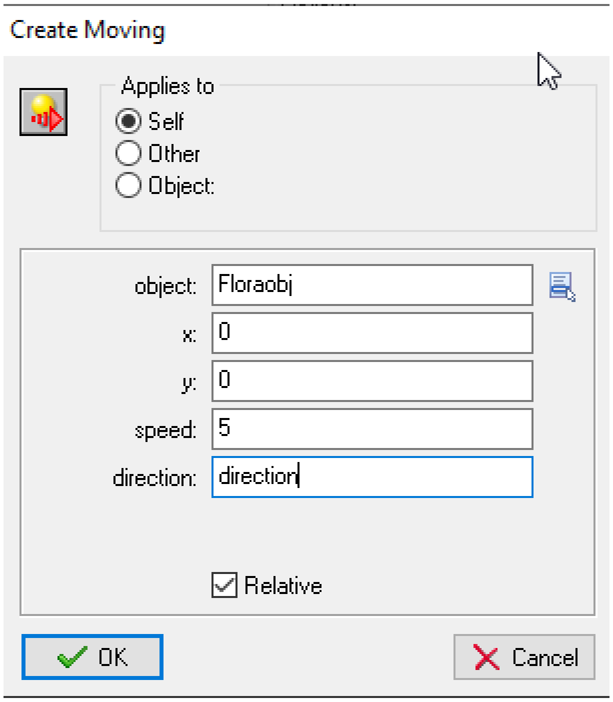
Adding a moving instance (8.1)

1. Add an event to the monsterobj keyboard press Spacebar.

2. For the action, click menu1 and drag the create moving object icon into the desktop.



3. For the object select your flora object. Click the relative check box to put the flora object next to the monster. For the speed set 5, and for the direction type **direction**.



Create Moving

Applies to
● Self
○ Other
○ Object:

object: Floraobj
x: 0
y: 0
speed: 5
direction: direction

☑ Relative

4. Run the program.

Adding Scripts (8.1)

There are two ways to control objects in GameMaker. One way, illustrated by the previous examples, is to use the GUI controls to define object behaviors. The problem is that most other game engines use scripts. GameMaker includes a scripting component that allows objects to be controlled by scripts.

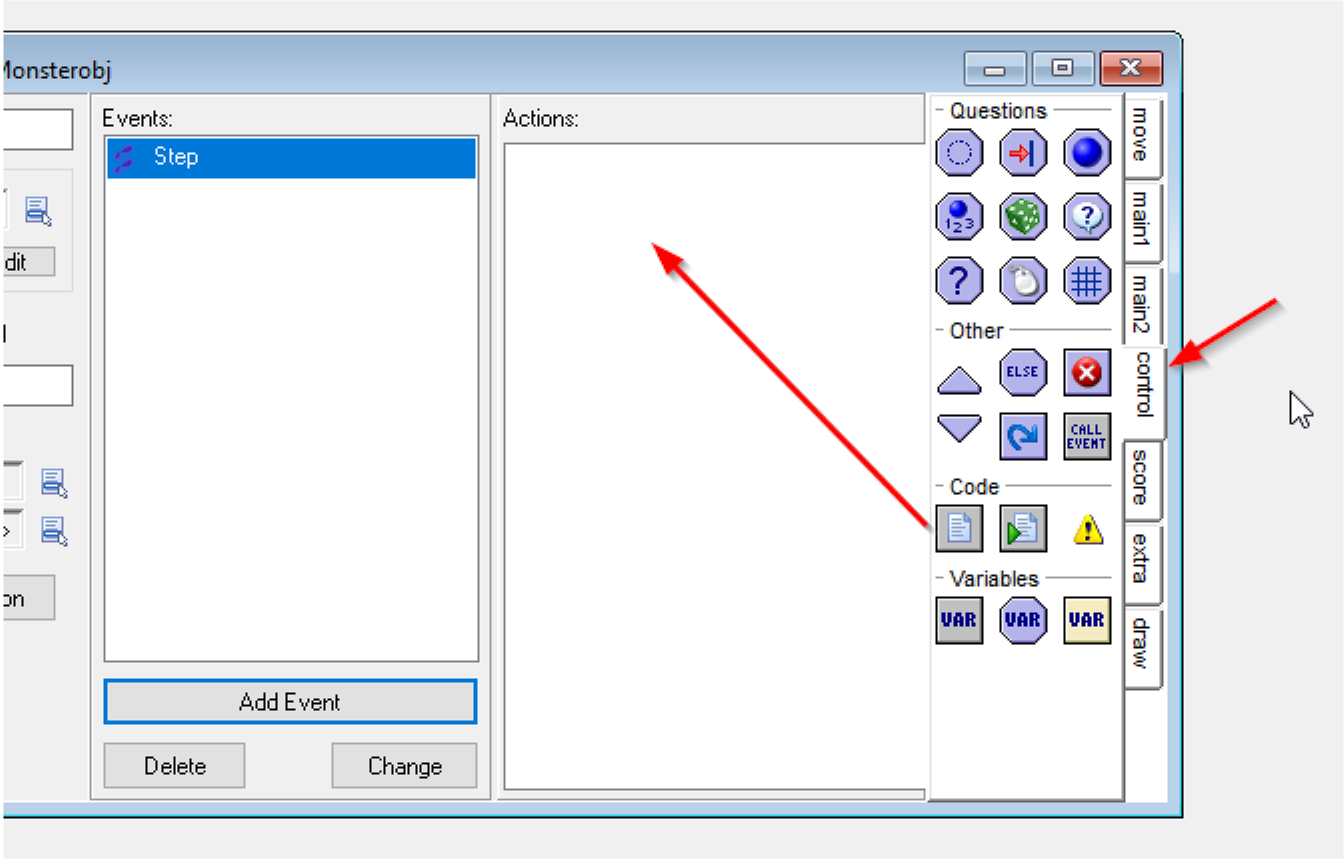
1. Open the GameGUI file and click Save As and save this with the name GameScript

2. Double click the Monster Object to edit the object.

3. The list of Events and Actions will have to be removed. For each event in the list select the event and click the Delete button to remove the event.

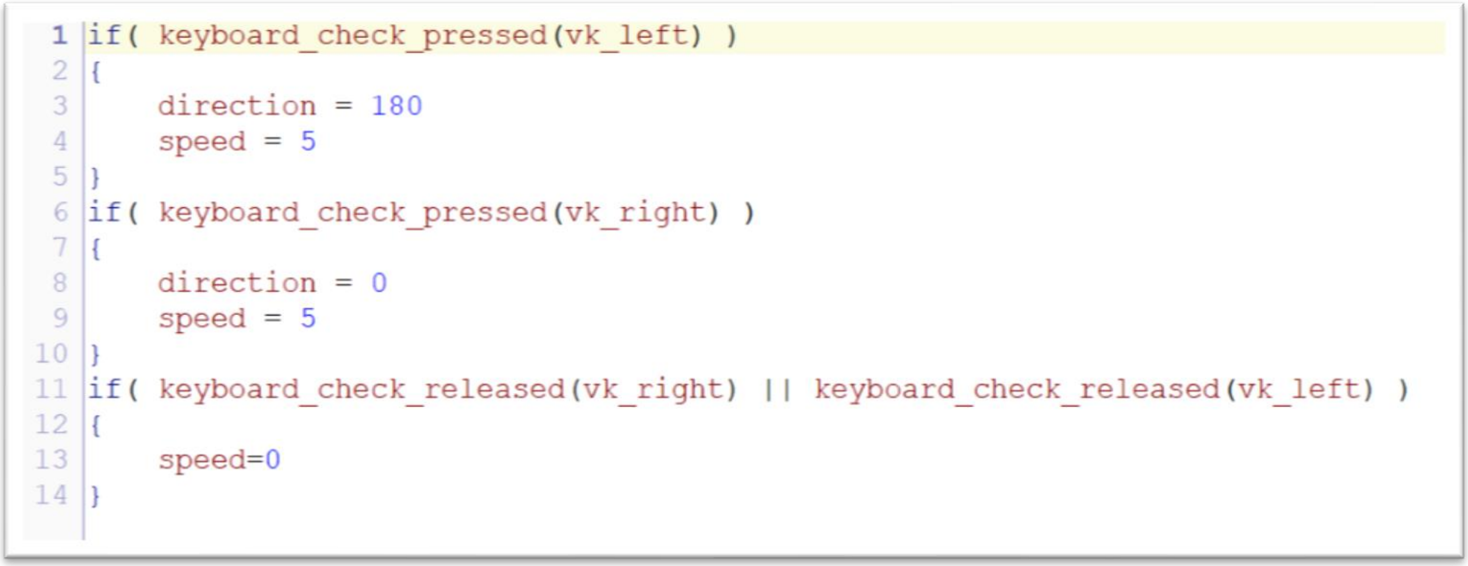
4. Click the Add Event button and choose Step button and choose Step. The step event will happen each time through the game loop.

5. Click the Control tab and drag the Script icon into the action area:



A scripting editor window shows up. You will add scripting commands to this window. One of the things to remember is that the scripts in this window will execute each time through the game loop.

6. In the script editor window enter the following script:



```
if( keyboard_check_pressed(vk_left) )
{
    direction = 180
    speed = 5
}
if( keyboard_check_pressed(vk_right) )
{
    direction = 0
    speed = 5
}
if( keyboard_check_released(vk_right) || keyboard_check_released(vk_left) )
{
    speed=0
}
```

This script performs the following three checks:

- If the left key is pressed then the object direction is set to 180 and the object speed is set to 5
- If the right key is pressed then the object direction is set to 0 and the object speed is set to 5
- If the left or right key is released then the speed is set to 0

The scripting language contains a set of IF statements and uses a set of built-in methods to perform actions. The structure of the IF statement is:

```
if(  condition )
```

where the condition can be either true or false. If the condition is true then the statement or statements directly following the if() statement will be executed . If the condition is false then the statements directly following the if() are skipped over. The statements following the if are enclosed in a pair of { } symbols.

```
if( condition )
{
        statements to perform if condition is true
}
```

The condition in this case is a method built into the game engine that checks the status of the keyboard. The key check method is:

```
keyboard_check_pressed(vk_left)
```

This method reads the keyboard and returns a true or false value. If the left arrow key (which is named “vk_left”) is being pressed then a true value is returned. The if statement containing this method registers a true value and executes the code contained in the { } block.

The code in the if() block consists of two commands, direction = 180 and speed = 5. Both of these commands assigns a value to the direction variable and the speed variable. These two variables are built-into the game engine and control what direction an object is moving and how fast the object is moving.

The direction variable can be set to a number indicating the degree of movement. Zero is to the right, 180 to the left, 90 is up and 270 is down. The speed variable can be set to the number of pixels the object is moving per second.

The entire command is:

```
if( keyboard_check_pressed(vk_left) )
{
        direction=180
        speed = 5
}
```

The direction and speed are set when the left arrow key is pressed. The same code can be used to check the right arrow key only the key (vk_left) is replaced by the right key (vk_right) and the direction is set to 0.

If the left or right keys are pressed then the direction and speed of the object is changed. However, when these keys are released (not pressed) the object speed needs to be reset to zero. This is done by the method:

```
if( keyboard_check_released(vk_left)  ||  keyboard_check_released(vk_right) )
{
        speed=0
}
```

This if() command is checking two conditions, releasing the left or right arrow keys. To check two conditions in the same if statement you have to use the conditions and/or. That is, checking on two methods means that both method A and method B are both true or either method A is true or method B is true. Inside an if() statement the and condition is represented by && and an or condition is represented by ||.

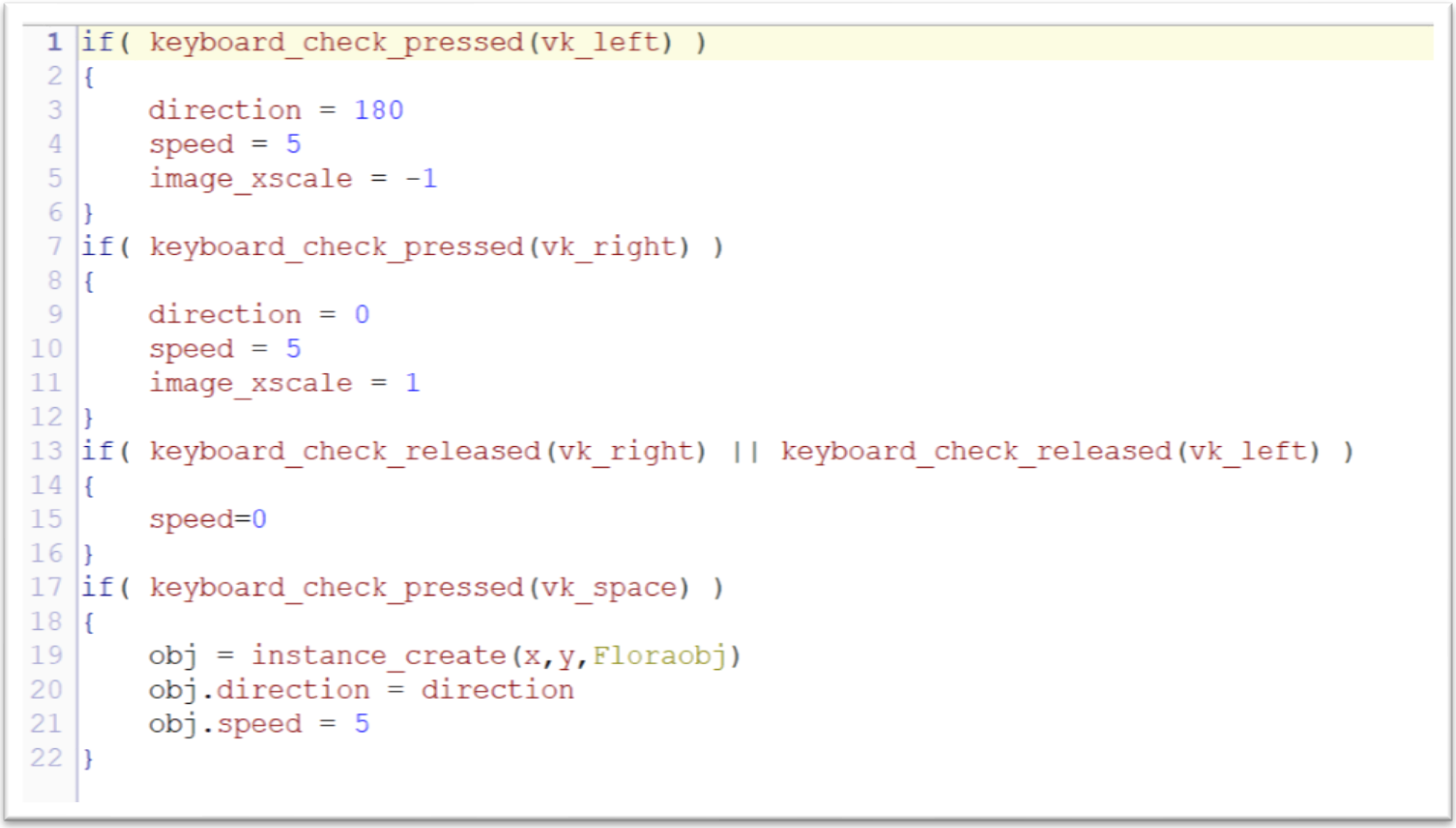
In this case, the if() command is checking to see if either the left arrow key or the right arrow key has been released. If either of these conditions are true then the speed is set to zero.

7. Click the check box to save the script and run the game. Notice that the object can be moved left and right.

Editing the Script (8.1)

While the object can be moved left/right the actual sprite always facing one direction. GameMaker has a sprite property which can flip the image. This property is image_xscale which can be set to 1 to preserve the default direction or -1 to reverse the default position.

1. Edit the script so it looks like:



```
if( keyboard_check_pressed(vk_left) )
{
    direction = 180
    speed = 5
    image_xscale = -1
}
if( keyboard_check_pressed(vk_right) )
{
    direction = 0
    speed = 5
    image_xscale = 1
}
if( keyboard_check_released(vk_right) || keyboard_check_released(vk_left) )
{
    speed=0
}
if( keyboard_check_pressed(vk_space) )
{
    obj = instance_create(x,y,Floraobj)
    obj.direction = direction
    obj.speed = 5
}
```

The property image_xscale is set to -1 when moving left to flip the image and set to 1 when moving right to set the image back to the default. The keyboard check pressed condition is set to check to see if the space bar is pressed (vk_space). If the spacebar is pressed, an object is created with the instance_create(x,y,Floraobj) command. Notice how this object is saved in a variable named obj. The direction and speed for this object is then set.

2. Run the game.

The next thing will be to create a moving instance of

GameMaker Studio 2.0

When you start GameMaker 2.0 you are given the option to create a new Drag and Drop project or a new Scripting project. We will do both.

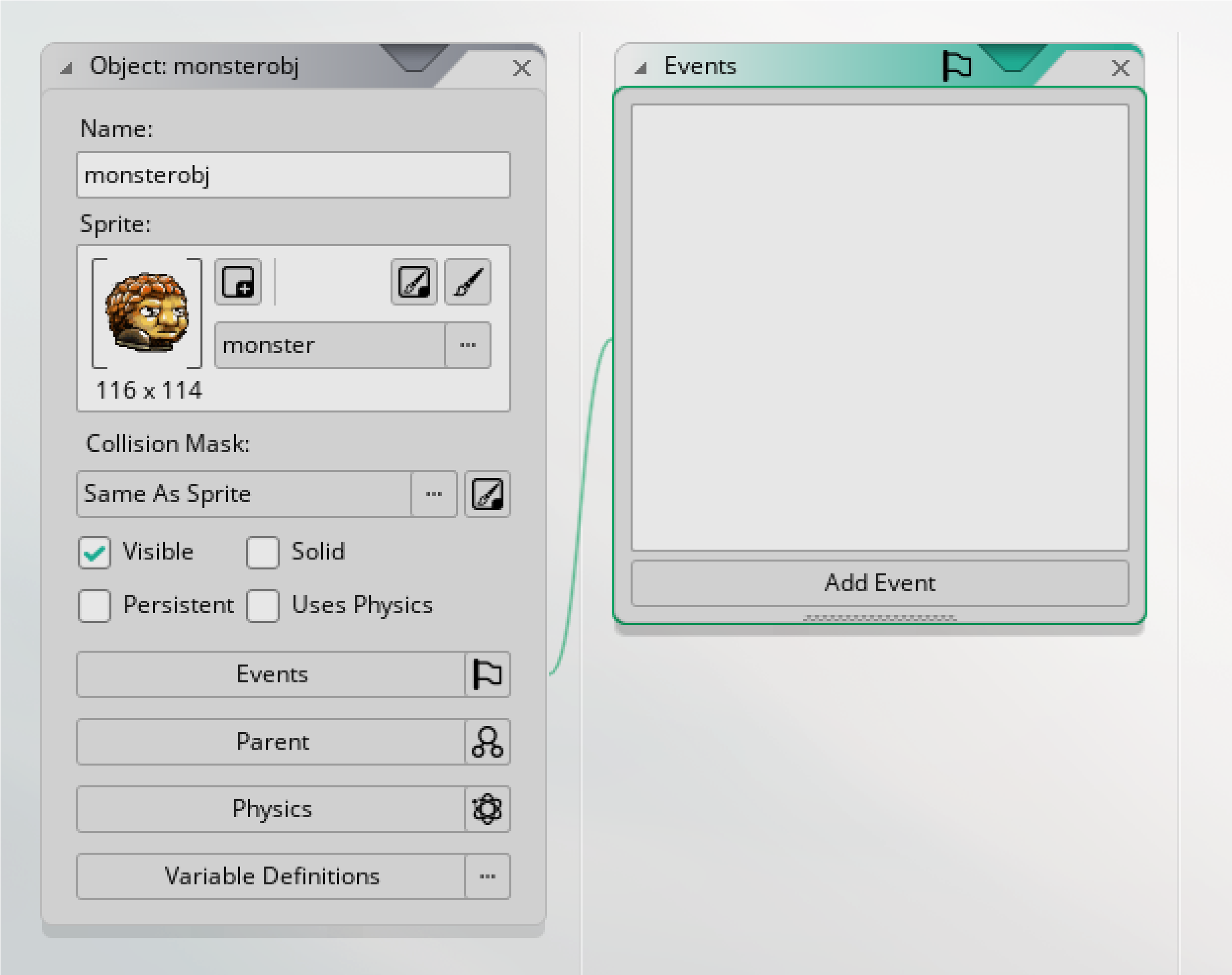
1. Start GameMaker 2.0, select Create a new Project, and choose Drag and Drop. Name the project ObjectGUI.

2. On the right side of the screen are the resource areas (version 8.1 has these on the left). Right click Sprites and choose Create Sprite from Images. Choose bg_4_02.jpg for the game background sprite.

3. Repeat this step and import the following images located in the animation-set folder or the icons128 folder. monster-walk-b.gif, door1.png, flora3.png. Name these (background, monster, door, and flora).

4. Create objects (using the sprites) for the Monster, the door, and the flora. Name these objects monsterobj, doorobj, and floraobj.

5. Double click the Monster object. You see the edit windows



Object: monsterobj

Name: monsterobj

Sprite: monster

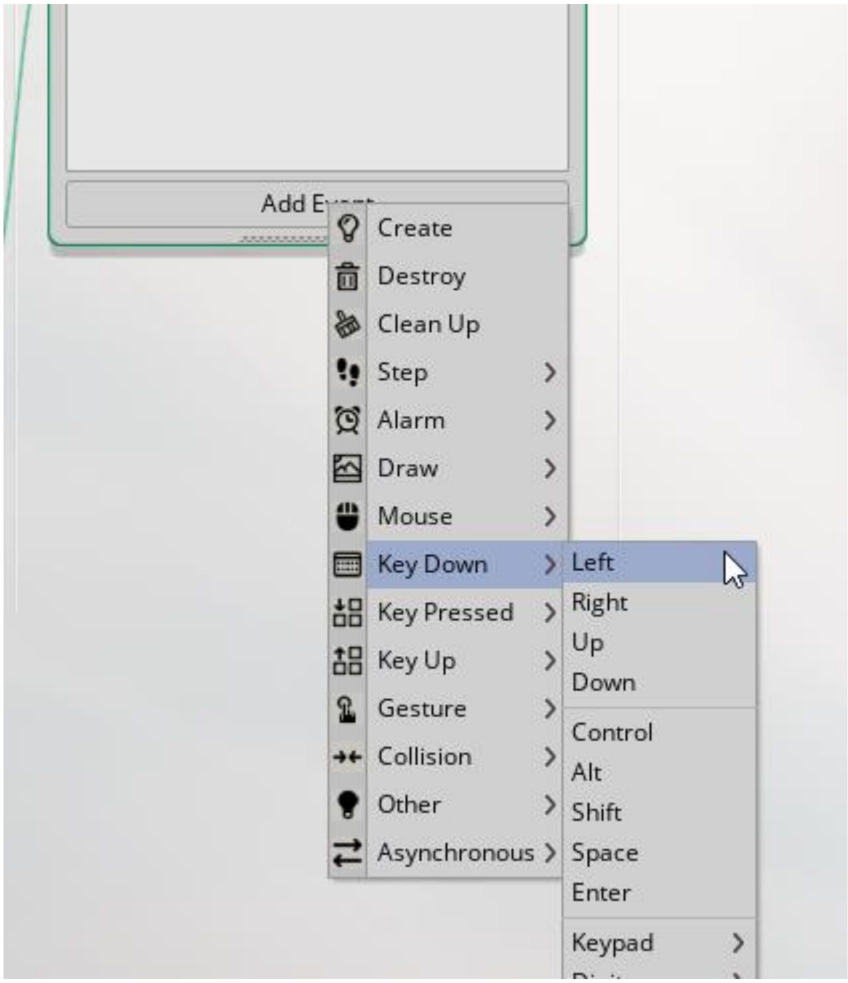
116 x 114

Collision Mask: Same As Sprite

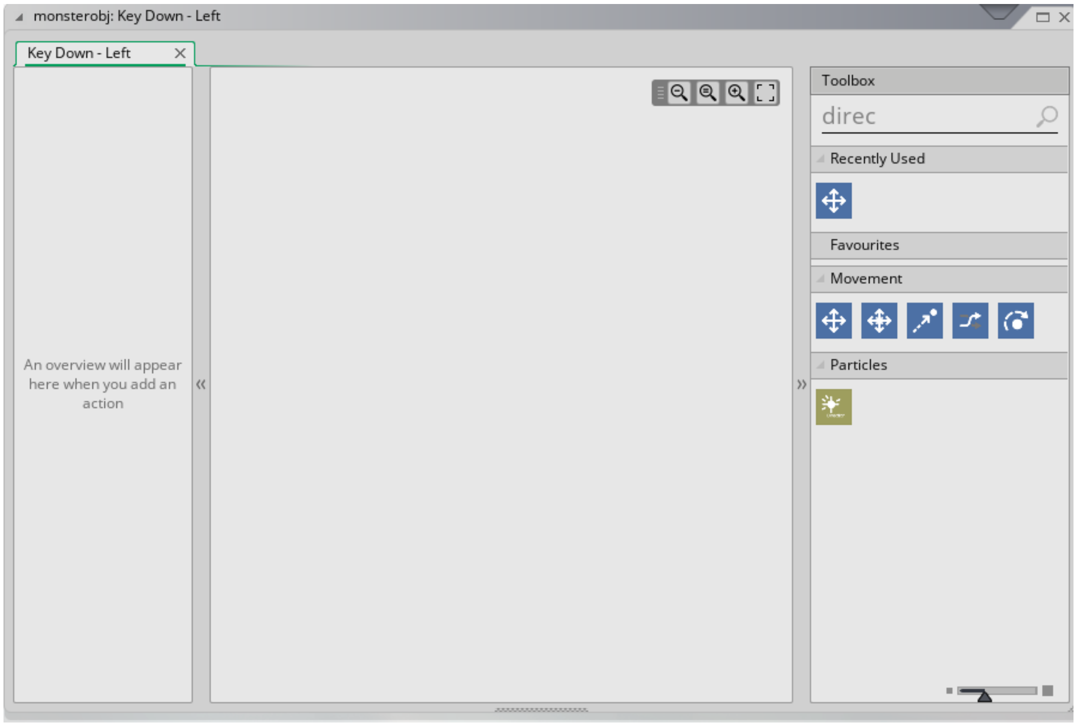
☑ Visible ☐ Solid

☐ Persistent ☐ Uses Physics

6. Click the Add Event button and choose Key Down/ Left

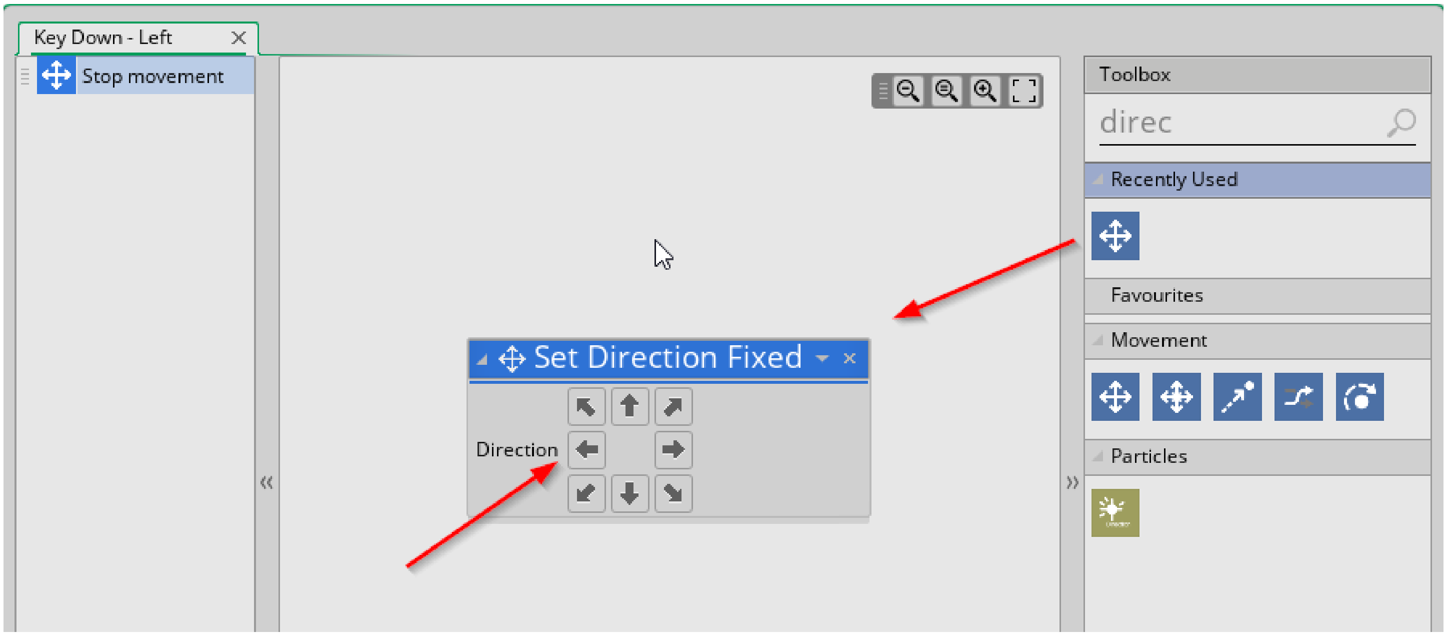


7. In the list of actions to perform dialog box type the word Direct. This lists the commands that involve setting the direction for moving for the object.



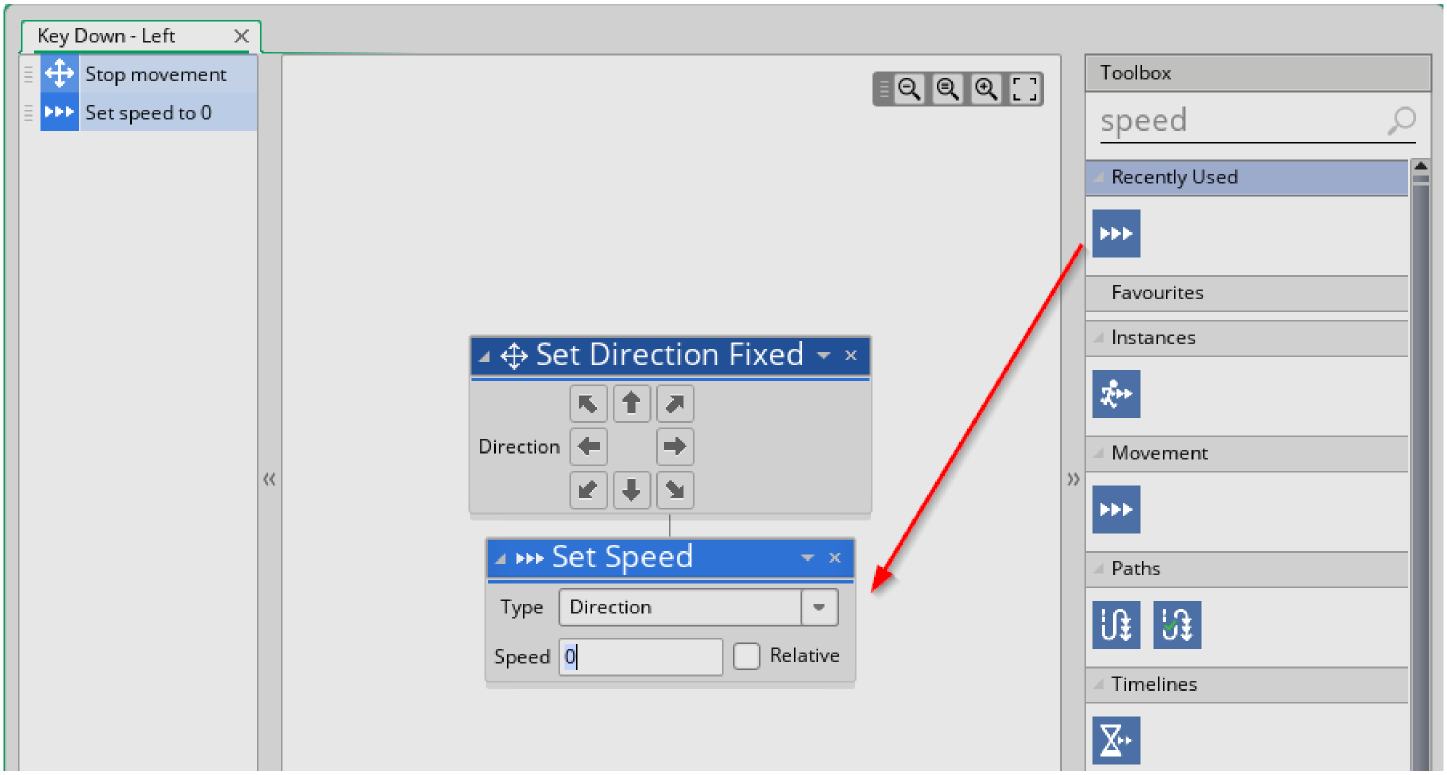
An overview will appear here when you add an action

8. Drag the box with the four sided arrow into the middle area and click the left arrow button.



You will next set the speed.

9. Type the word speed in the toolbox search box and drag the button with three arrows below the Set Direction Fixed block.



Type: Direction

Speed: 0

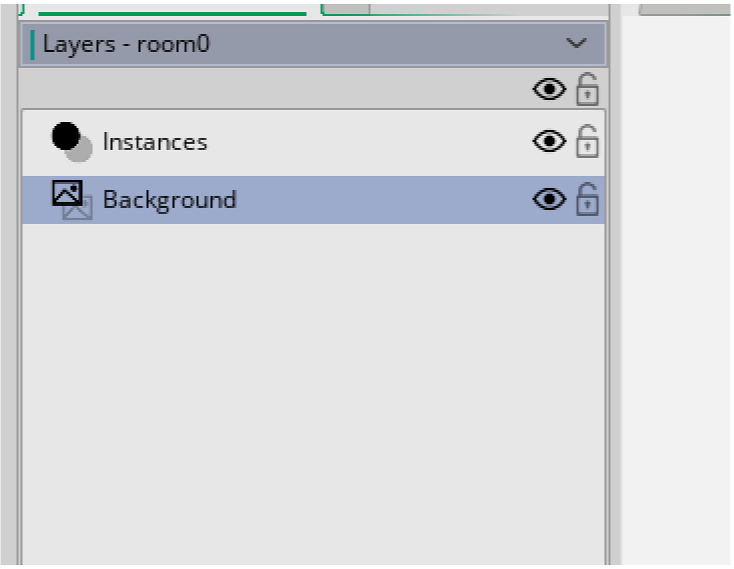
10. Enter the speed value 5

11. Click the Add Event button and, using the same steps, set the direction and speed for moving right.

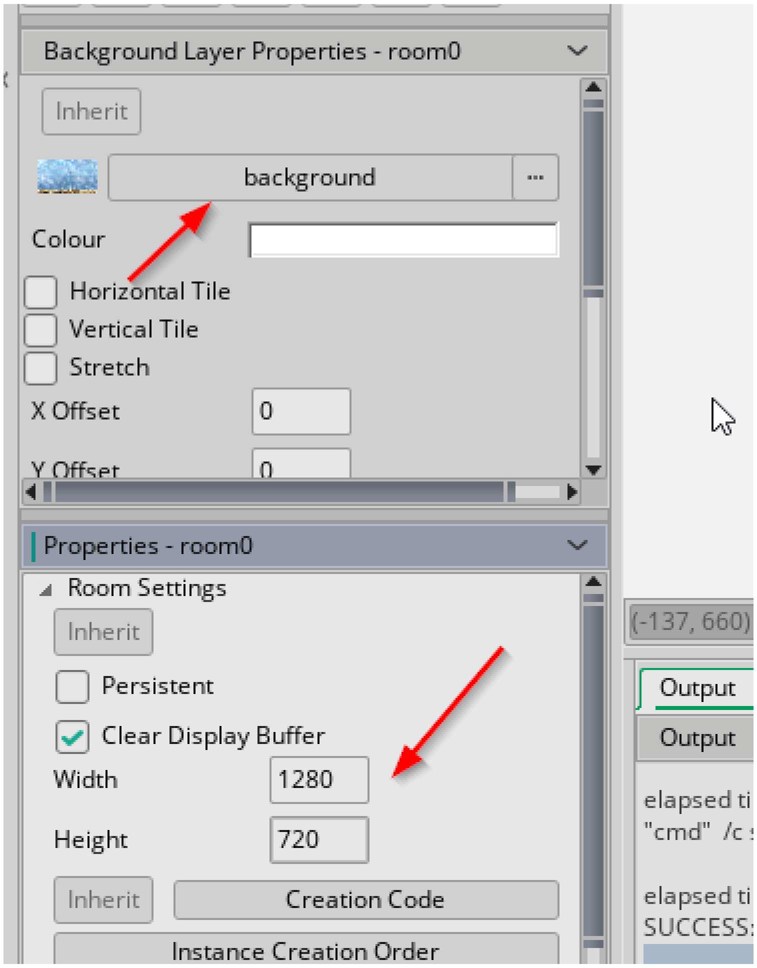
12. Using these same steps, add the Key Up for the left key and set the speed to zero and add the Key Up for the right key and set speed to zero.

13. Double click the room object in the Room area.

14. Click the Background option



15. In the lower window choose the background sprite and set the size to 1280 by 720 (the size of the background sprite).



background

Colour

□ Horizontal Tile

□ Vertical Tile

□ Stretch

X Offset 0

Y Offset 0

□ Persistent

☑ Clear Display Buffer

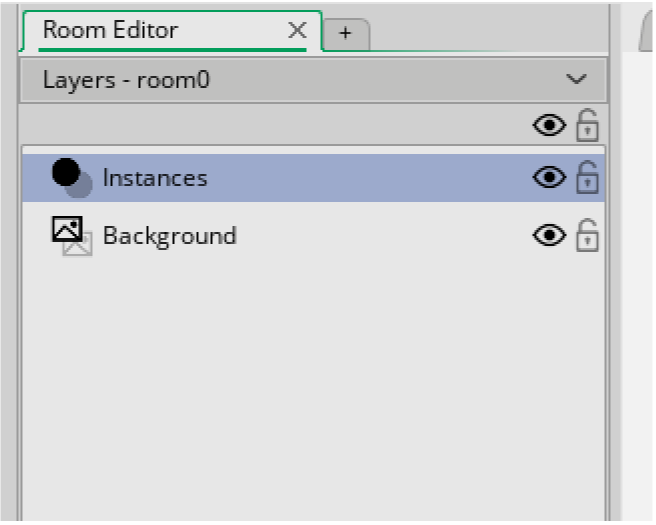
Width 1280

Height 720

```
elapsed ti
"cmd"  /c 

elapsed ti
SUCCESS:
```

16. For the room, click the Instances option. You can only add objects to the room when the Instances option is chosen.



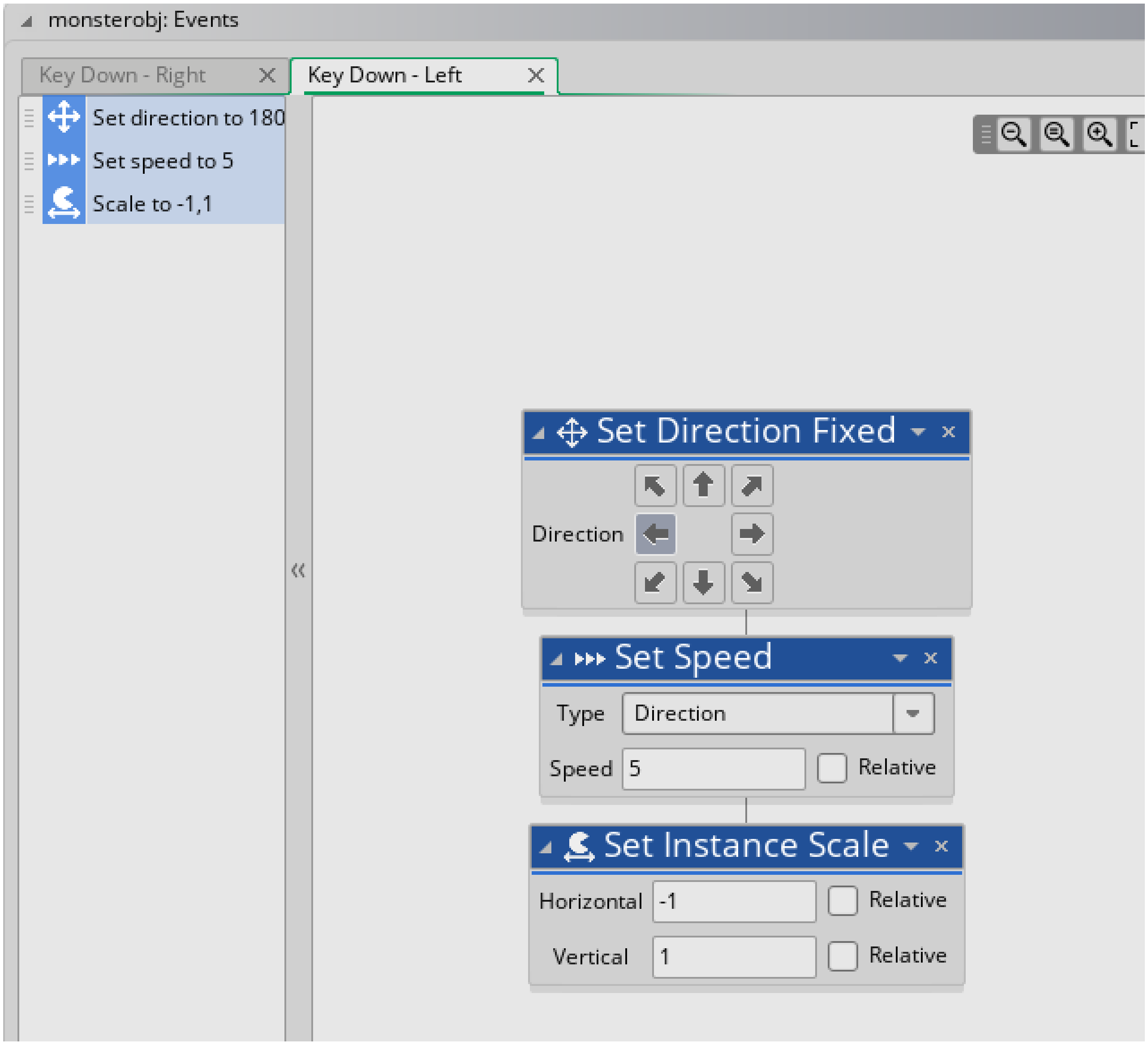
17. Drag your Monster and Door objects into the room. Run the program. If your monster walks to slowly you can edit the sprite and change the speed of the animation.

18. Double click the Monster object.

19. Double click the Keydown Left event.

20. In the Action dialog box search for Scale. You will see an icon button with a double sided arrow.

21. Drag this icon into the action area and it will add a Set Instance Scale option. Enter -1 for the horizontal option.



Set direction to 180
Set speed to 5
Scale to -1,1

Direction

Type Direction

Speed 5 □ Relative

Horizontal -1 □ Relative

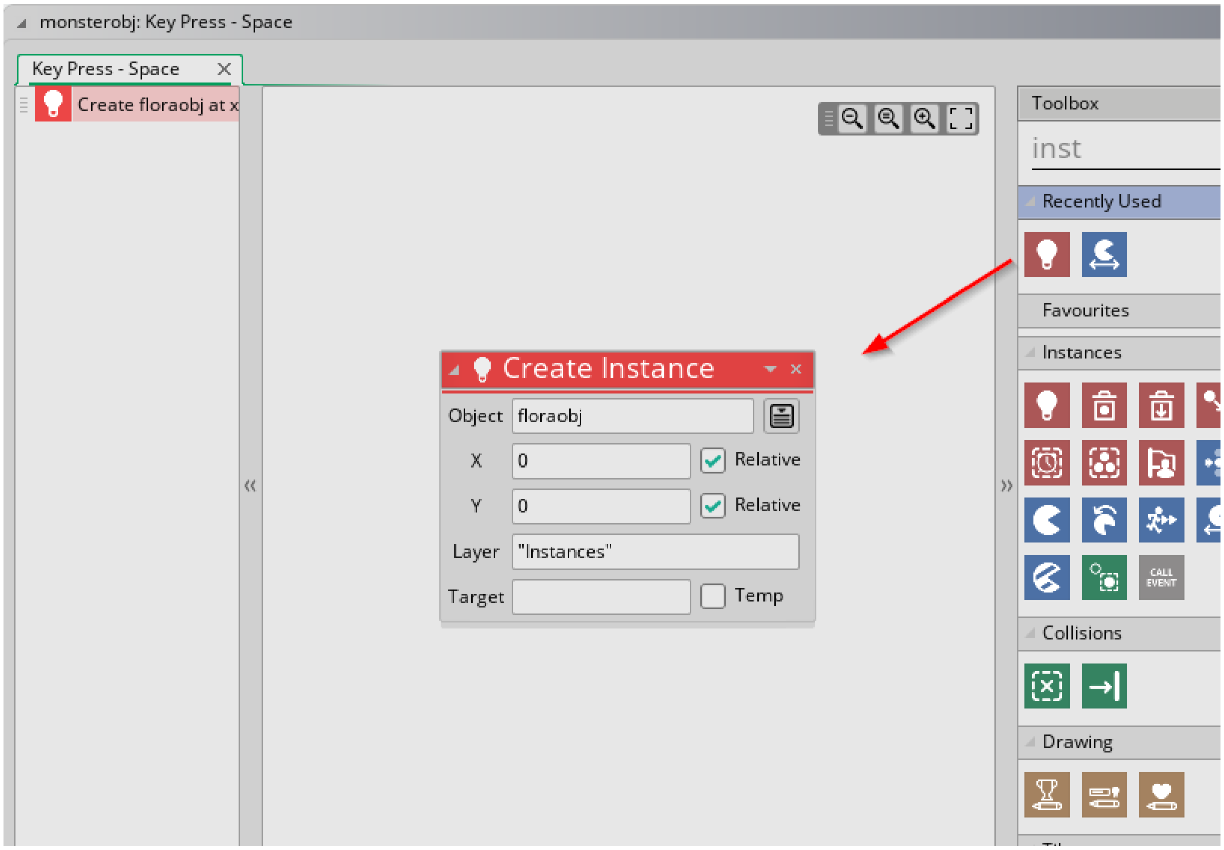
Vertical 1 □ Relative

22. Add the Set Instance Scale for the press right arrow key. When your program the image will reverse if you move to the left or right.

23. Add a Key Press event for the space key.

24. In the toolbox search area type Create. This shows a red icon button with a lightbulb.

25. Drag this icon into the Action area.

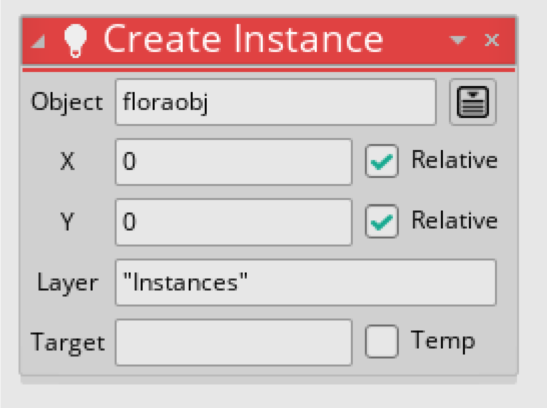


Create Instance

Object	floraobj	
X	0	☑ Relative
Y	0	☑ Relative
Layer	"Instances"	
Target		□ Temp

This will be used to create an instance of the flora object.

26. Choose the floraobj object and click the relative check box for both x and y.



Create Instance

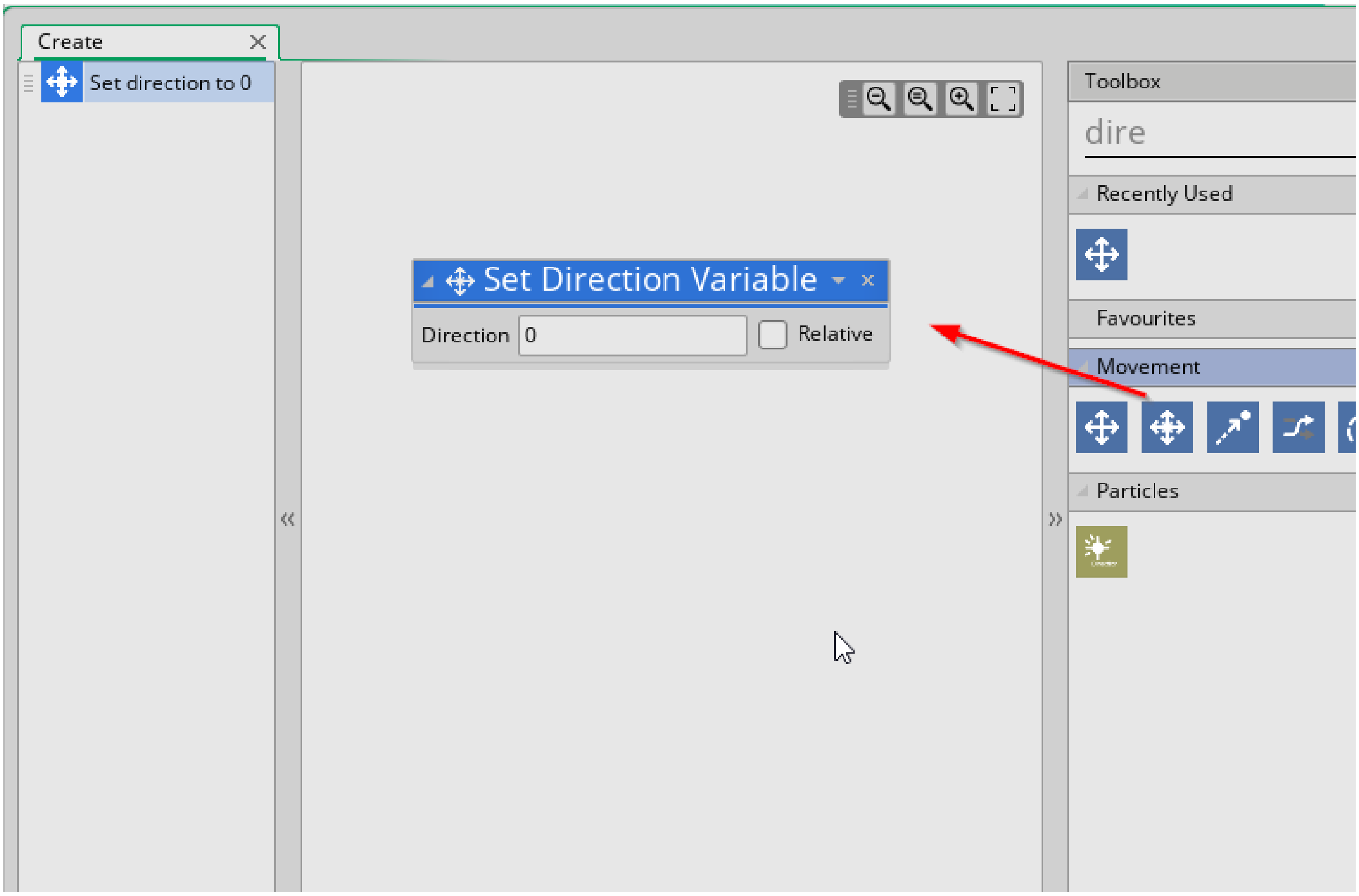
Object	floraobj	
X	0	☑ Relative
Y	0	☑ Relative
Layer	"Instances"	
Target		□ Temp

You now need to set the direction and speed of a floraobj when it is created.

27. Double click the Floraobj.

28. Click the Add Event button. You want to add the Create event. This happens once when an object is created. You need to set a speed for this object and a direction. The direction will be the direction of the Monster object.

29. In the toolbox type Direct to show the direction variables. Drag the set direction variable into the action area:



30. In the Direction variable enter **monsterobj.direction**. This will retrieve the direction value from the monster. This means that the direction of the flora created when a spacebar is pressed will take it's value from the direction of the monster.

31. Search for speed and add it to the object and set the speed to 5.

32. Run your game. The monster will throw an object if the spacebar is pressed.

GameMaker 2.0 Scripting

1. Create a new GameMaker scripting project.

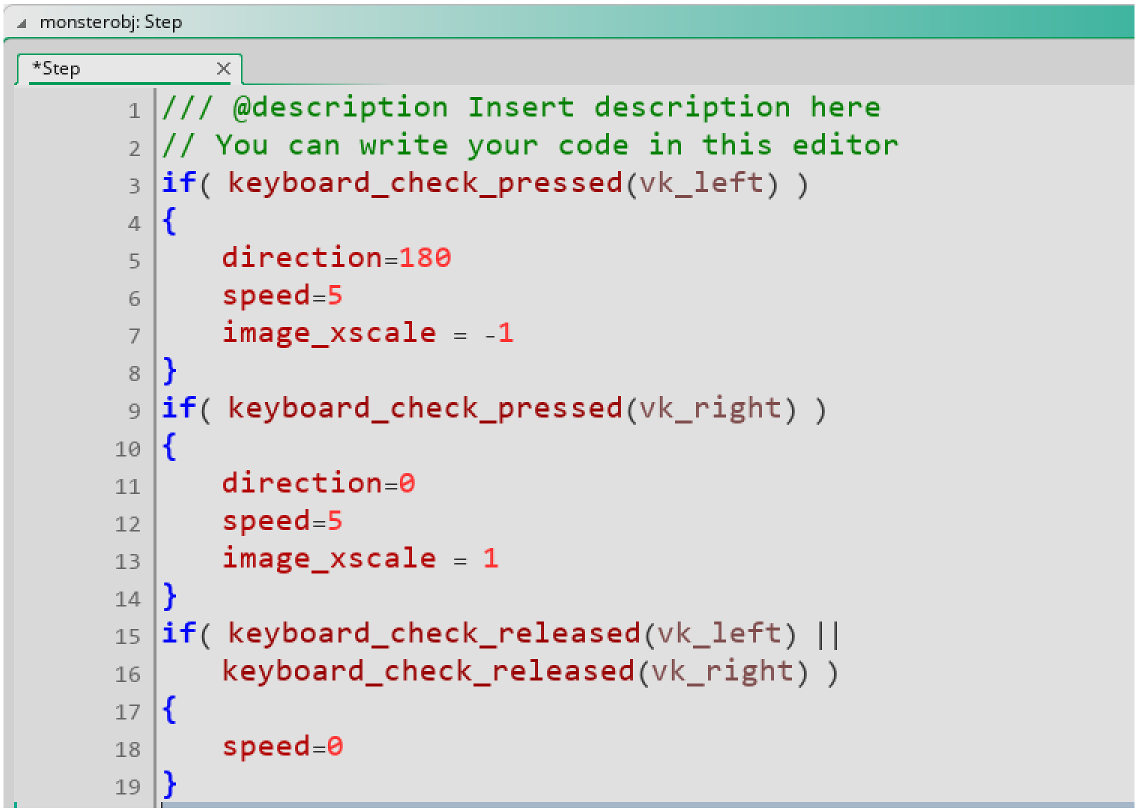
2. Follow the GUI steps to import sprites, set the room with the background, and creating objects.

3. Add the Door object to the room and the monster object to the room.

4. Double click the Monster object, add an event, and choose Step.

5. You see the Code window.

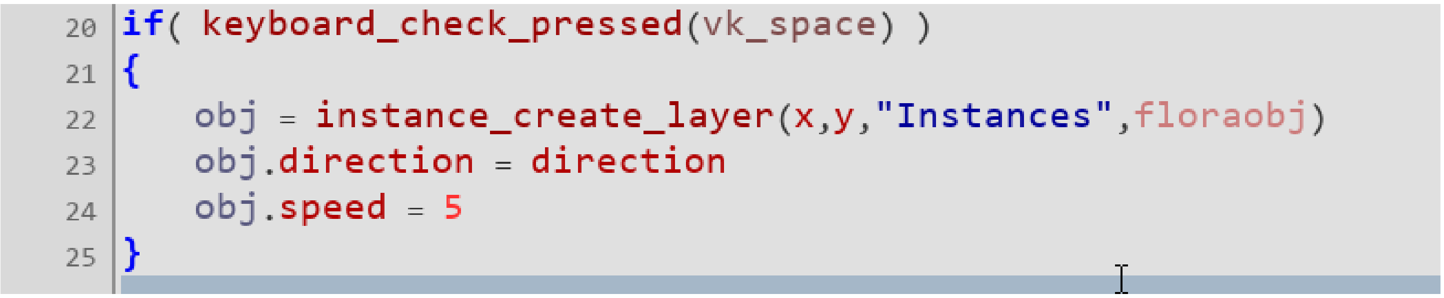
6. In the code window type the script:



```
/// @description Insert description here
// You can write your code in this editor
if( keyboard_check_pressed(vk_left) )
{
    direction=180
    speed=5
    image_xscale = -1
}
if( keyboard_check_pressed(vk_right) )
{
    direction=0
    speed=5
    image_xscale = 1
}
if( keyboard_check_released(vk_left) ||
    keyboard_check_released(vk_right) )
{
    speed=0
}
```

7. Run the program. The monster can be controlled by the left/right keys.

   To add the moving object you would edit the code and add:



```
if( keyboard_check_pressed(vk_space) )
{
    obj = instance_create_layer(x,y,"Instances",floraobj)
    obj.direction = direction
    obj.speed = 5
}
```