What is an Event?

Events are things that happen. In programming, some of the events that can occur are clicking or moving the mouse, typing a key on the keyboard, or other user interaction.

When you run a program, you might press a key on the keyboard, move the mouse, or click with the mouse. These are examples of events. When these events occur, nothing will happen unless you tell the program to **listen** for these events and what to do when these events occur: how to **handle** the event. If we want to execute some statements when one of these events occurs, we have to tell Alice to "listen" for that event. We do this by adding a listener. We tell it what event to listen for, and what function to execute when that event occurs.

Listening for events, and **handling** them lets you make your program more interactive by allowing the user to control the action using the keyboard and the mouse. You will also use a timer event to make a dog wag its tail continuously or add events that happen at random intervals.

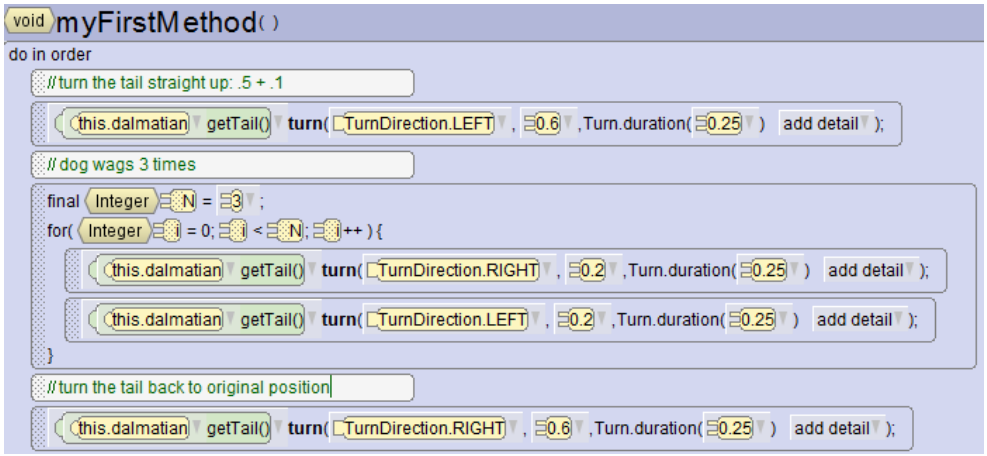
After adding an event listener, that event will trigger the execution of a procedure or function. Programmers refer to this as "Event Handling."

You will be able to make interactive games and programs that allow the user to control an airplane or other vehicles with the keyboard and use the mouse to make selections.

You will also detect when an object collides with another object.

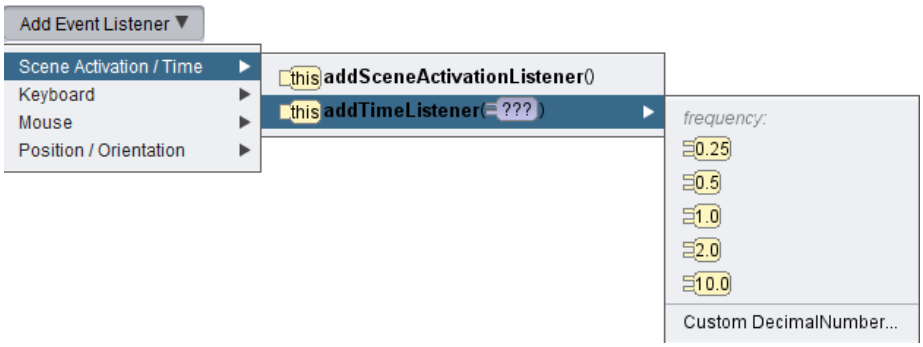
Dog Wags Tail

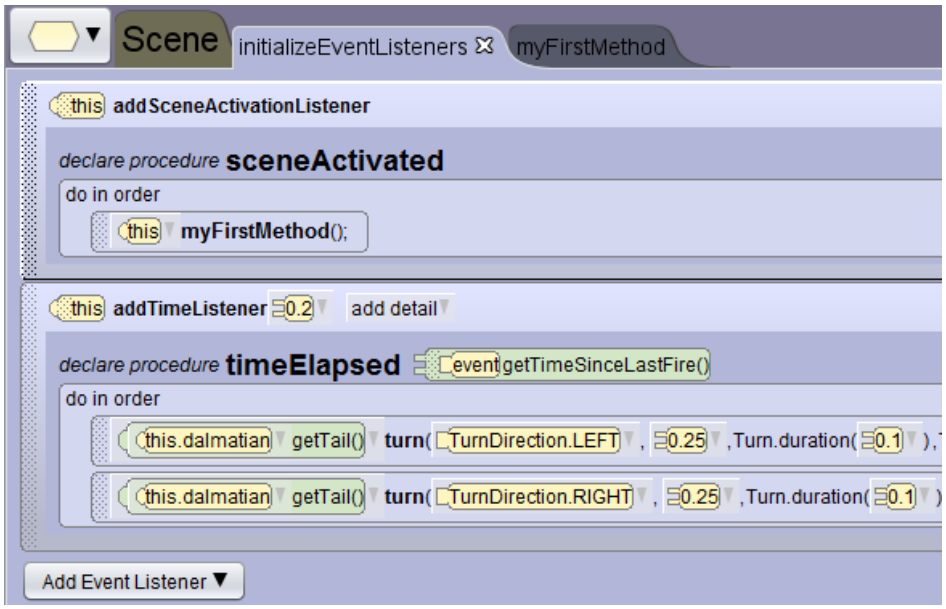
Here is code in myFirstMethod for the dog to wag his tail. The dogs tail points straight up, it moves back and forth 3 times, then returns to the starting position. The problem is that nothing else can happen until the god finishes wagging his tail.



Dog Wags Tail Continuously

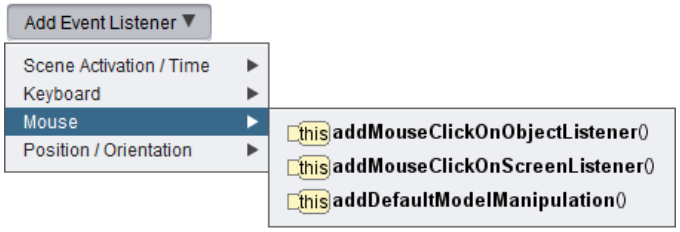
To make the dog wag his tail continuously, click on the initializeEventListener tab, then add a Time Listener. The dog will wag his tail back and forth each time the timer goes off. By setting the timer to 0.2 and the duration of each of the two tail movements to 0.1, he wags his tail continuously. The code in myFirstMethod will execute at the same time. You could also create a procedure where the dog wags his tail and call the procedure in the timer event.



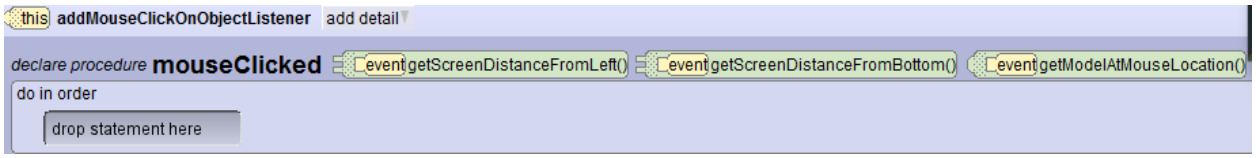


The Mouse

In the next example we will have some animals. I would like the animal I click on to do something. Click on Add Event Listener and select Mouse, addMouseClick OnObjectListener().



When we add the mouse event listener, we will see the procedure. At the top are three functions that will give use information about the event. We will be able to drag these functions into the code. We would like to know if getModelAtMouseLocation (the object that was clicked) is the bear.

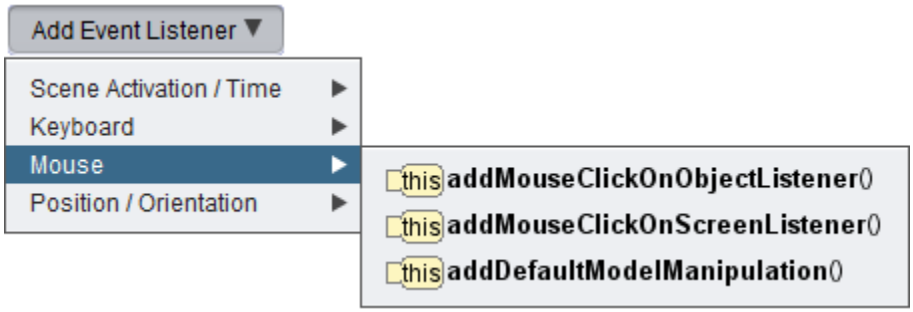


We will add an If command and select true initially. Then we will replace that to compare two SThings. We will replace the two SThings with the bear, and then drag the getModelAtMouseLocation function in to get this code:

Now (finally) we can add statements to make the bear do whatever we want to do when we click on him.

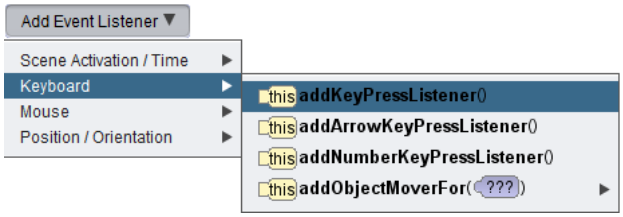
Drag and Drop

If we select Add Event Listener, then select Mouse, addDefaultModelManipulation, we will be able to drag and drop objects in our scene.

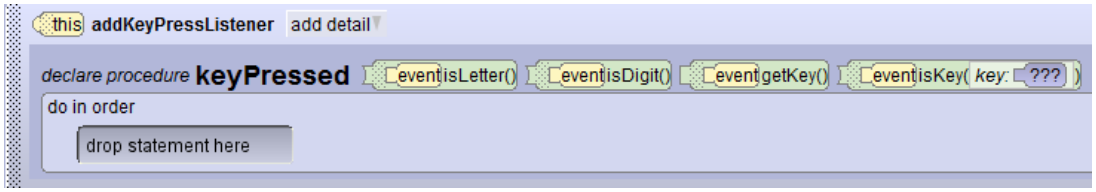


The Keyboard

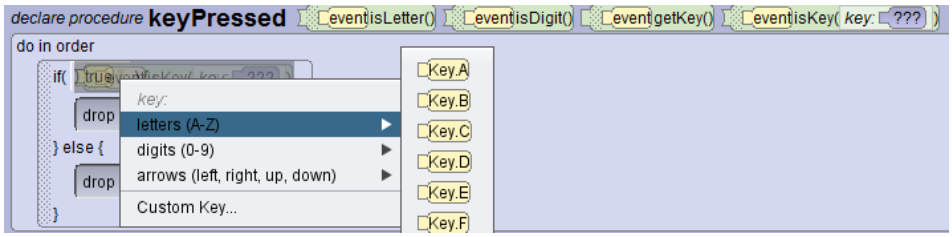
In the next example we will set up the scene with two characters: a bear and a moose. When we press the letter "B on the keyboard the bear does something. When we press the letter "M" the moose does something.



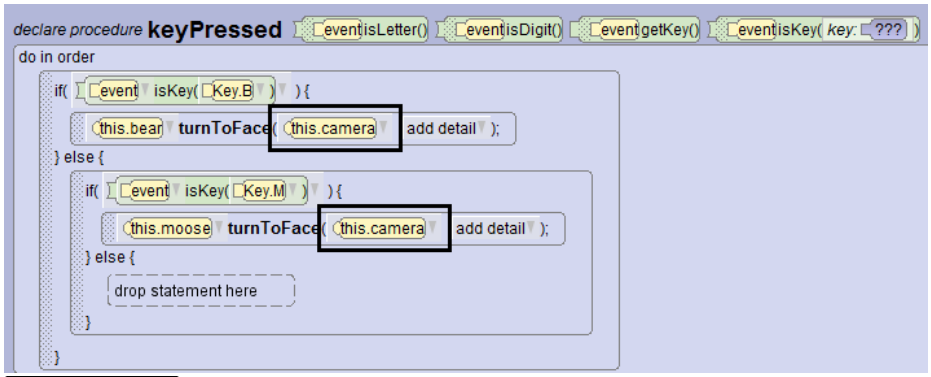
Select Add Event Listener, Keyboard, addKeyPessListener(). You will see the procedure for the keyPressed event and 4 tiles representing functions that return information about the event:



Drag an if tile into the procedure. Make it true initially, then drag the isKey function in to replace true.



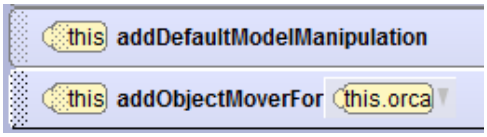
The isKey function lets you select a letter of the alphabet from A to Z; a digit 0-9; the arrow keys, and custom key. We will select the letter "B" to represent the bear. You can create the code shown here, or any other code you want.



The Orca and Carp Game

We can do a lot with what we have learned so far. We're going to create a simple game which you can then use to make something much more interesting. The movie is set up with the ocean floor and we have added an orca and a carp.

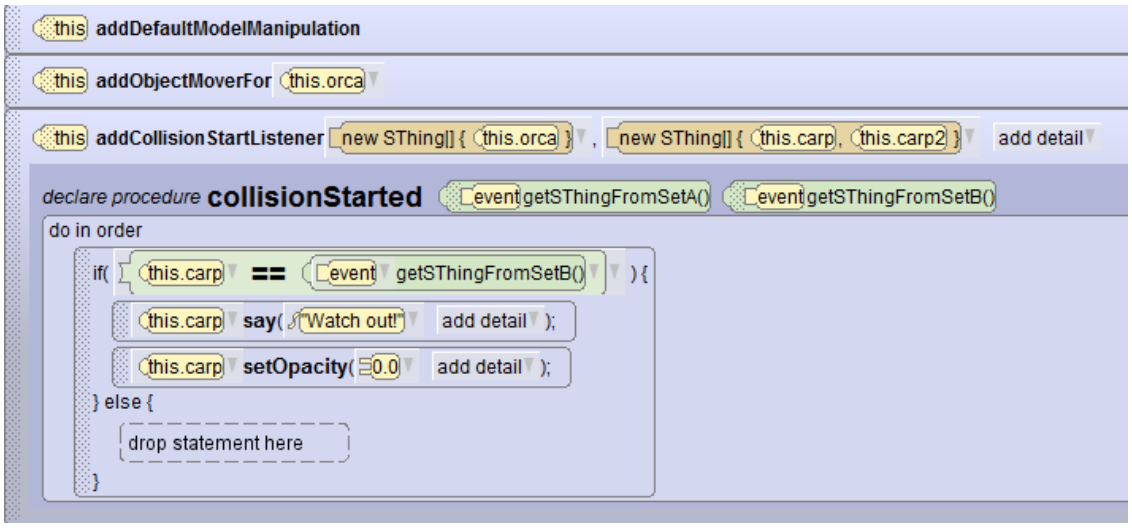
In event listeners add the two procedures addDefaultModelManipulation and addObjectMover for the orca. The Default Model Manipulation will let you drag and drop any object that you click. The addObjectMoverFor (orca) will let you turn the orca and move it with the arrow keys.



The left arrow key will make the orca turn to HIS left, and the right arrow key will make him turn to HIS right. The up arrow will make him move forward and the down arrow key will make him move backward. (If you addObjectMoverFor both the carp and the orca, they will both move or turn when you press the arrow keys.)

If you would like to be able to move the orca up and down, you could add a key press listener and move the orca up when you press "Q" and down when you press "Z". *("Q" and "Z" are used in some games for up and down, but you could use any keys you want.)*

We are going to have the orca chase the fish and catch them. We need to know when the orca collides with a fish. The collisionStarted event requires 2 arrays as argumens. The first array is the list of things that caused the collision; the second array is the list of things it collided with.



Set up the code as shown in the illustration. Experiment with it to see how it works so far. Now you are ready to make a more interesting game.

- Add a point each time the orca catches a fish.
- Make the fish swim around randomly.
- After a fish is "eaten", give it a new position (maybe using a marker) so that it can swim into view as a new fish.
- Add a timer. When the timer goes off the game is over.
- Add some hazards: if the orca collides with one of those he loses points or the game is over.
- If a fish swims out of view of the camera, reposition it to swim into view again.
- Fish can swim into a cave to hide. If all of the fish swim into the cave, the game is over and the orca loses.
- Use your imagination to create an interesting game.

There are other events that you can experiment with. Try all of the different events that you see in the drop down list. What do they do? How could you use them to create a different game or movie. Enjoy!