ORACLE Academy

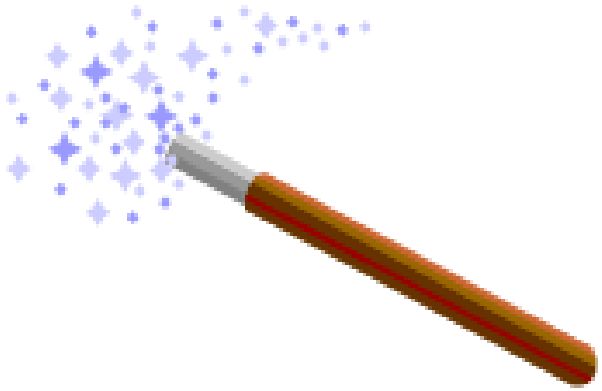
Using Expressions

What Will I Learn?

Objectives

In this lesson, you will learn how to:

- Create an expression to perform a math operation
- Interpret a math expression



Why Learn It?

Purpose

As you test and debug your animation, you will want to refine the movements of objects so they move and act exactly the way you intended.

Expressions can be used in your procedure's arguments that use math calculations and functions to further refine and perfect an object's movement.

Expressions

If you have taken a math course, you may recall mathematical expressions defined as a combination of values that, when arranged correctly, will result in a final value.

For example: 2+2=4

The expression is created using two values (2 and 2) and the operator (+). Combining the values with the operator result in the new value, 4.

Expressions (cont.)

Expressions are created in Alice 3 animations using built-in math operators. They are:

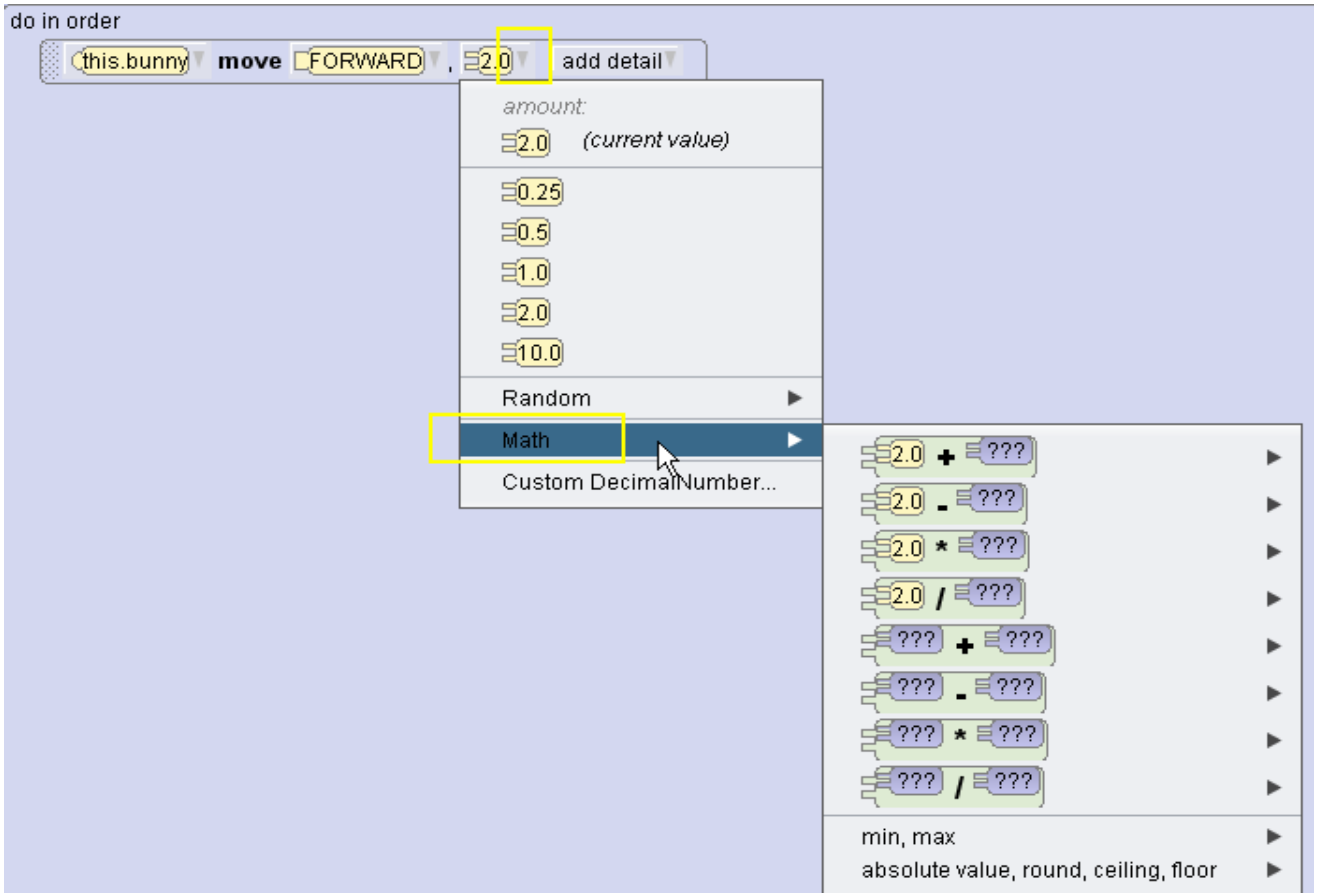
- Add (+)
- Subtract (-)
- Multiply (*)
- Divide (/)

Math operators are available in the same menu where you select values for the:

- Amount and Duration arguments in procedures
- Get Distance functions

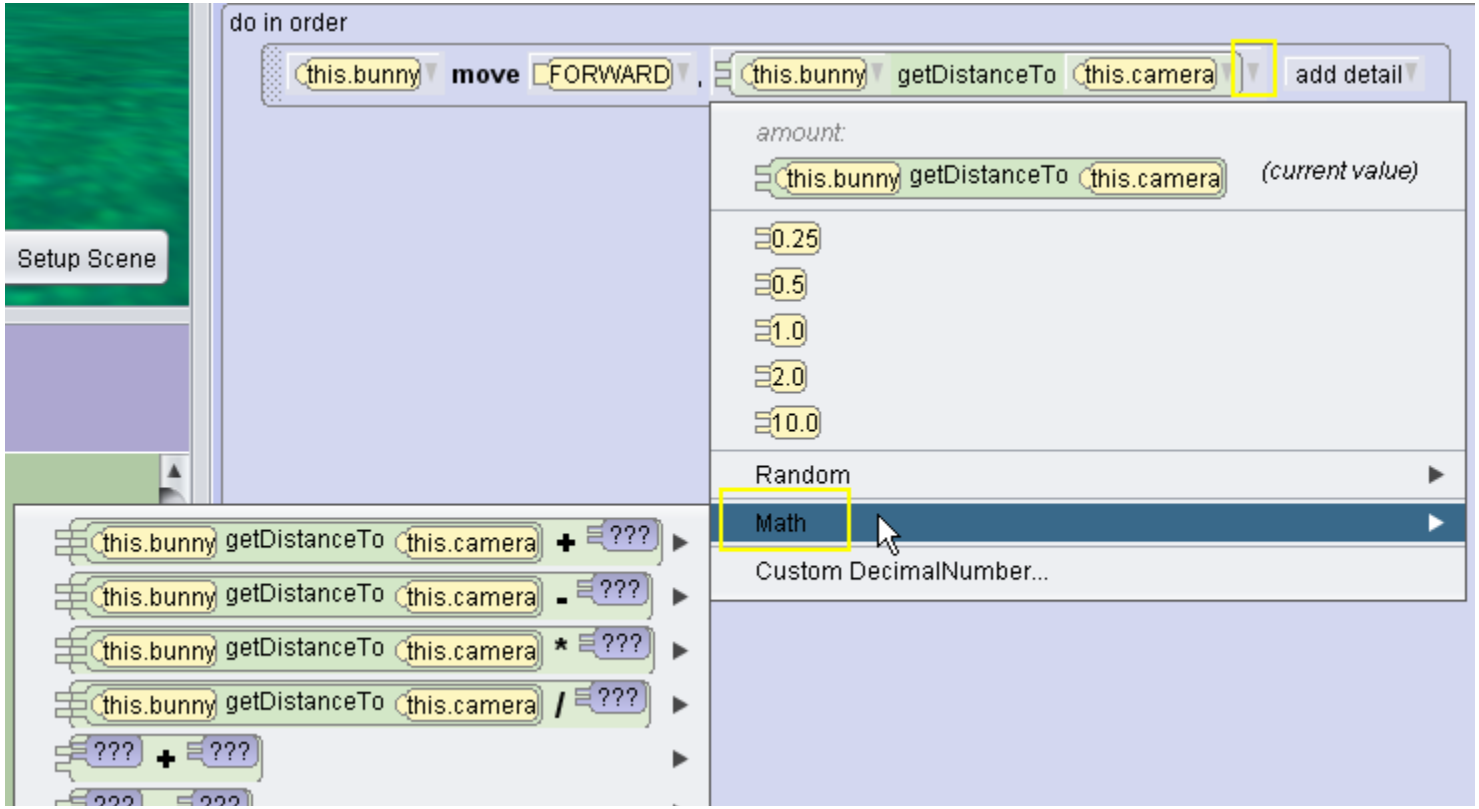
Expressions (cont.)

Select the Math option to view the math operators available for the distance argument.



Expressions (cont.)

Select the Math option to view the math operators available for the getDistanceTo function's argument.



Expressions (cont.)

How to use math operators to create an expression:

1. Summarize the problem
2. Identify timing or distance problem
3. Consider a solution
4. Consider an expression
5. Code the expression
6. Debug the expression

Expressions (cont.)

Problem: In Alice 3, the person object walks forward to a rock and stops in the center of the rock, rather than near the rock. This is because the getDistanceTo function calculates the distance to the center of the target object.

To make the animation look more realistic, the person should not stop in the center of the rock (because this would not happen in the real world!).

Expressions (cont.)

Problem Solving Steps

1. Summarize the problem
   - Person walks through a rock object, and stops at the rock's center.
2. Identify timing or distance problem
   - Distance
3. Consider a solution
   - Use getDistance function and reduce the distance traveled by some amount.
4. Consider an expression
   - Person getDistanceTo rock minus 0.5, 1 or 2, etc, (i.e.subtract whole number) OR
   - Person getDistanceTo rock minus getDepth from rock (i.e., subtracting with getDepth)

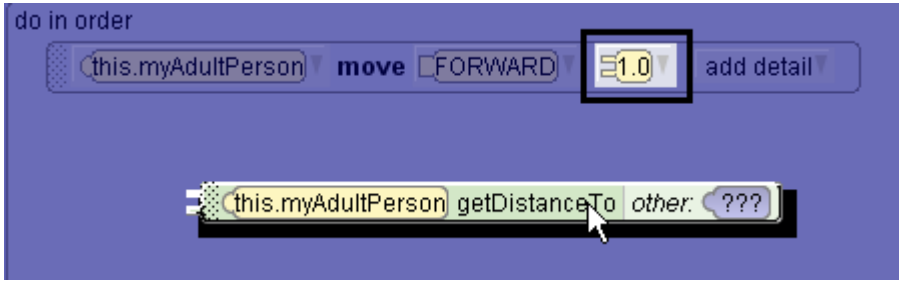
Expressions (cont.)

Code an expression to solve the distance problem.

Steps:

1. Drag a move procedure into the code editor (direction = forward, distance 1 meter).

2. Go to the Functions tab. Drag the getDistanceTo function onto the distance argument.



Expressions (cont.)

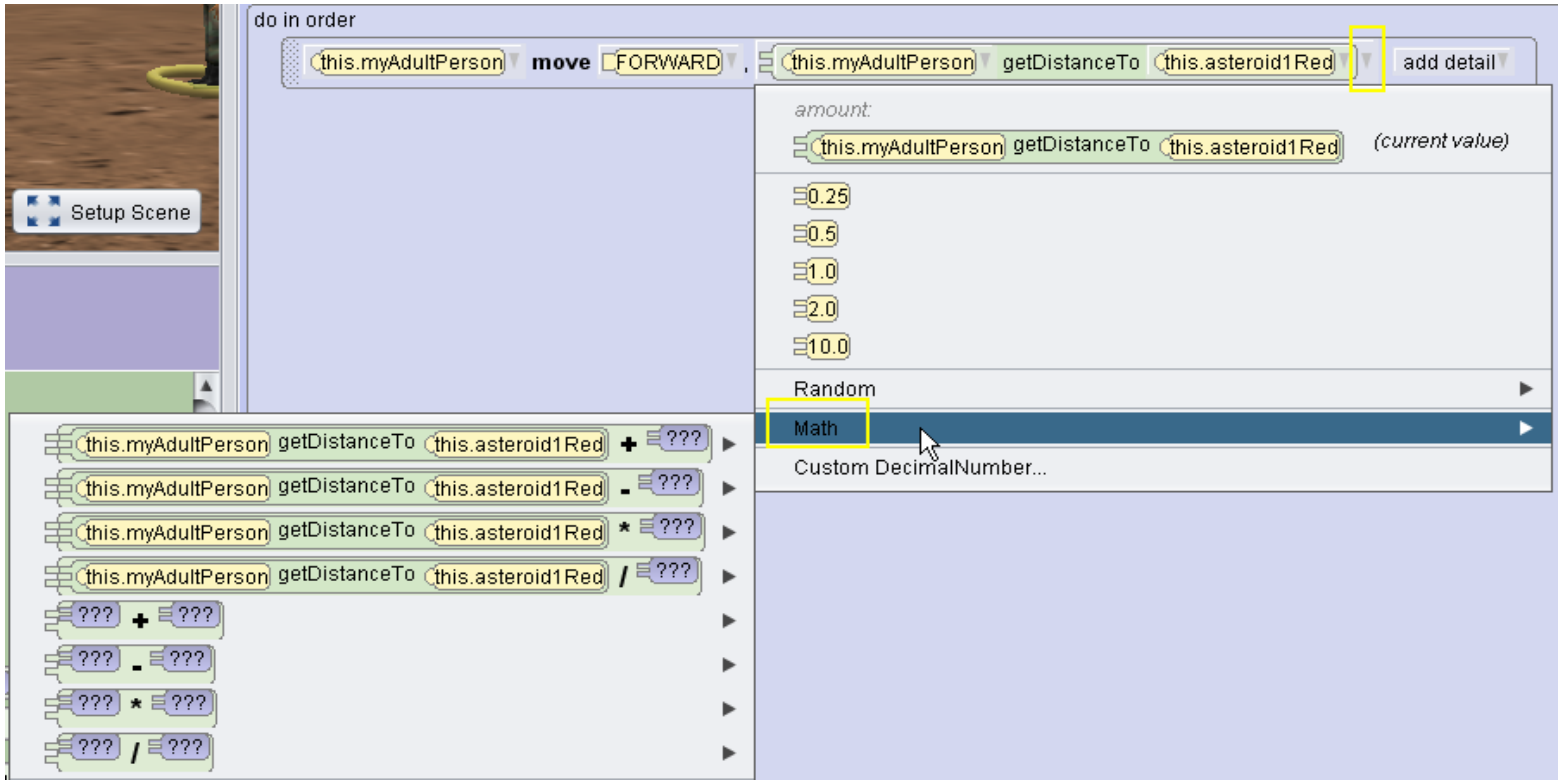
Steps (cont.):

3. Select the target object that the person should move to.

Expressions (cont.)

Steps (cont.):

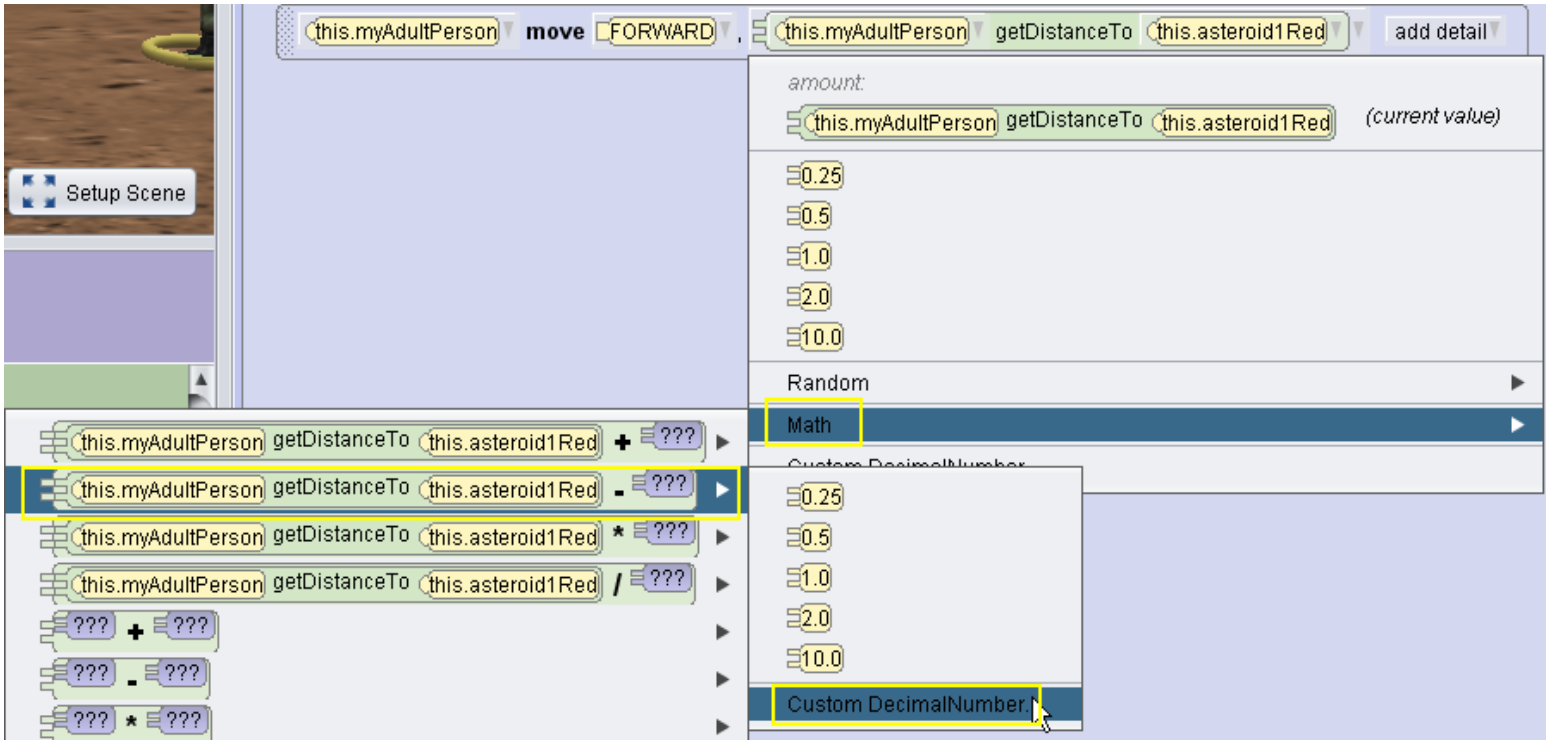
4. From the getDistanceTo tile, click the outer-most down arrow icon, then select Math.



Expressions (cont.)

Steps (cont.):

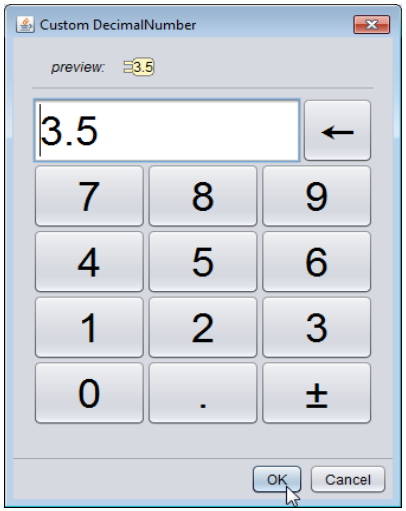
5. Select getDistanceTo - ???
6. Select a number to reduce the distance by, or select Custom DecimalNumber... to enter a number.



Expressions (cont.)

Steps (cont.):

7. Type in an amount to reduce the distance, then click OK.



8. Review the final expression.

Expressions (cont.)

During the testing and debugging process, you may need to adjust the value of the expression.

If you need to adjust the value to subtract from the distance argument, click the arrow next to the value, and select a new value.

Expressions (cont.)

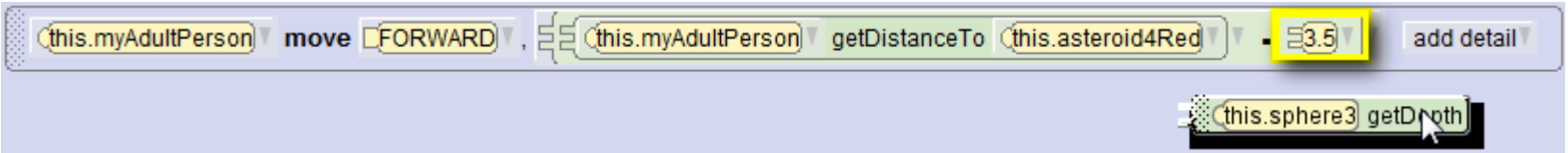
A more precise way to ensure that the moving object lands directly next to the target object without going to the center of the object is to subtract the depth of the target object from the expression.

This saves time so you don't have to guess at the distance, as well as repeatedly test and debug various values that you enter into the expression.

Expressions (cont.)

Steps:

1. From Functions tab, drag the getDepth tile onto the distance value, 3.5.

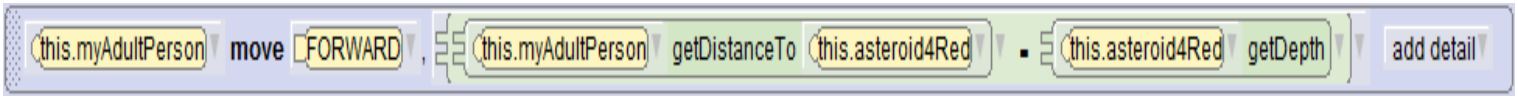


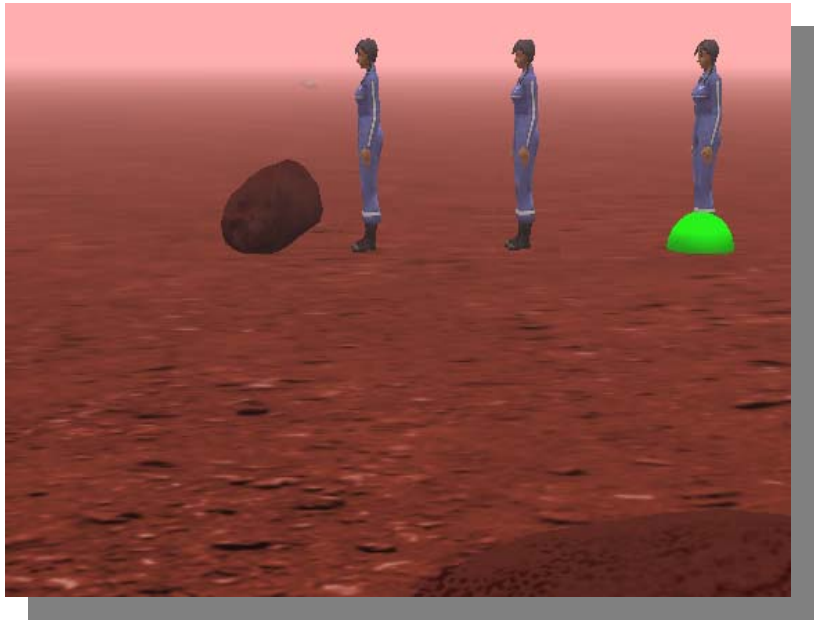
2. Select the target object (the Asteroid).

Expressions (cont.)

Steps:

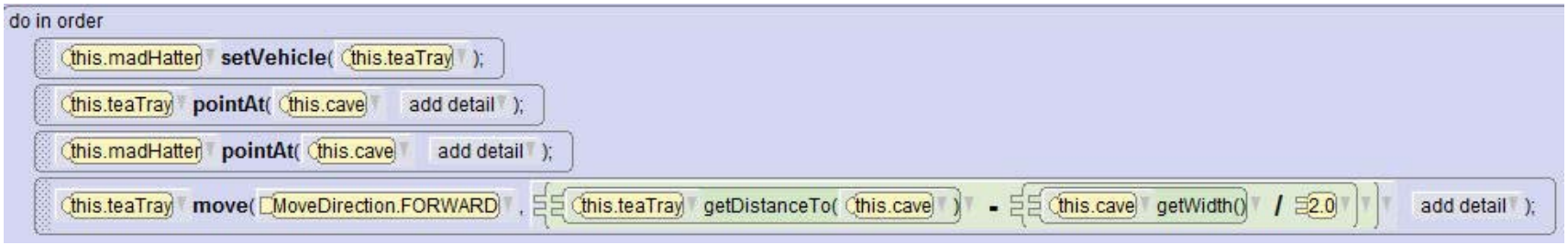
3. Test and debug the final expression by running the animation and adjusting the values of the expression accordingly.





Intrepret an Expression

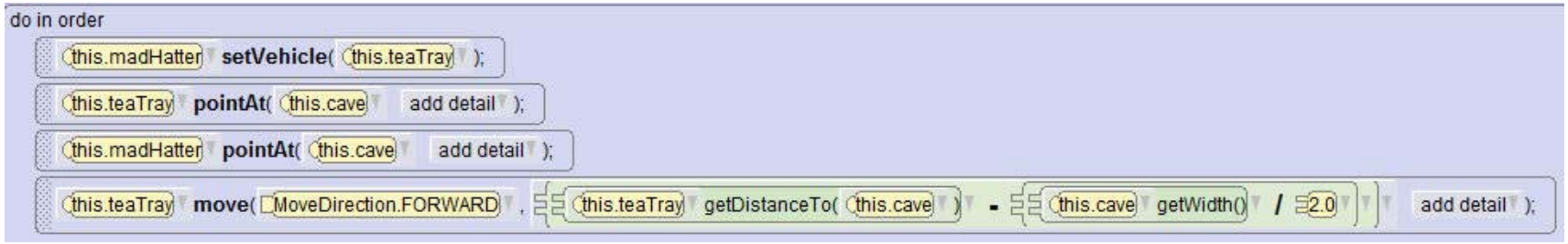
Expressions can be interpreted by reading an expression from left to right. Consider this example:



Begin evaluating the expression by recognizing the instances that are specified. In this example we see that there is a teaTray and cave instance. We can also determine by reading all of the code that there is also a madHatter instance involved in this animation. The madHatter instance is moving with the teaTray instance.

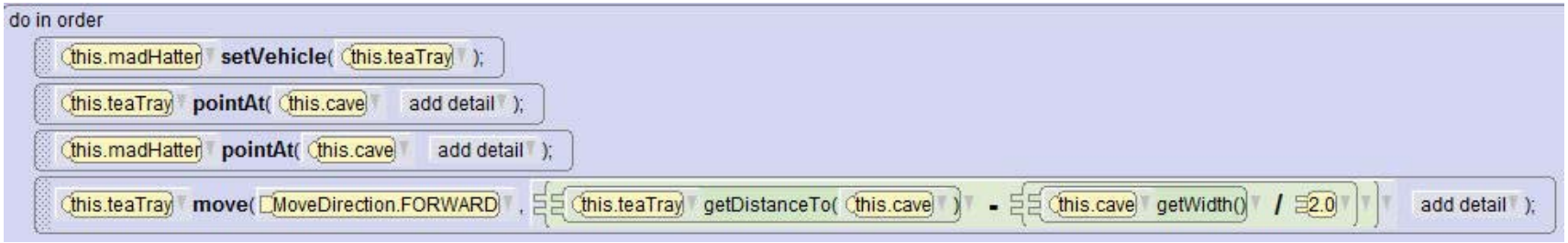
Intrepret an Expression (cont.)

Examine the visual associated with this expression. You can now visualize that the madHatter and teaTray instance are moving towards the cave. Will they go into the cave?



Intrepret an Expression (cont.)

The expression tells us that the teaTray is moving forward towards the cave. The distance between the teaTray and cave is being determined by the getDistanceTo function. The distance being moved is being reduced by half of the width of the cave (width divided by 2). The width of the cave is being determined using the getWidth function.



To interpret an expression you may need to draw a picture or write down the values you know before formulating the expression:

$$Z = x – (a / b)$$

Z = distance moved; x = distance from tray to cave; a = cave width; b = 2

Terminology

Key terms used in this lesson included:

Expression

Summary

In this lesson, you learned how to:

- Create an expression to perform a math operation
- Interpret a math expression

Practice

The exercises for this lesson cover the following topics:

- Create math expressions
- Interpret math expressions