Getting Started with Java Using Alice

Use Procedures and Arguments

Objectives

This lesson covers the following objectives:

- Toggle between, and describe the visual differences between, the Scene editor and the Code editor
- Locate and describe the purpose of the methods panel and the procedures tab
- Use procedures to move objects
- Add Java programming procedures to the Code editor

Objectives (cont.)

This lesson covers the following objectives:

- Demonstrate how procedure values can be altered
- Create programming comments
- Reorder, edit, delete, copy, and disable programming statements
- Test and debug an animation

Display the Code Editor

Click Edit Code to display the Code editor. The Code editor is where you program your animation.

Select Instance

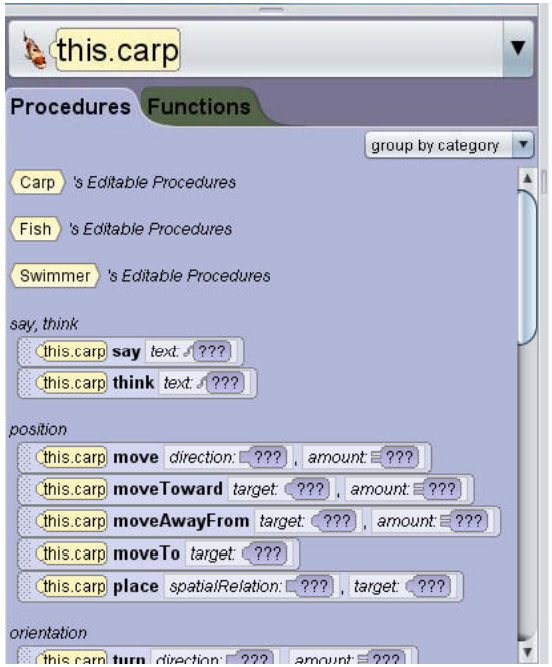
First, select the instance that you want to program. This ensures that you are creating a programming instruction for the correct instance.

Select an instance by clicking on the instance in the small scene window or by using the instance pull down menu below the small scene window.

Methods Panel

The Methods Panel contains two tabs:

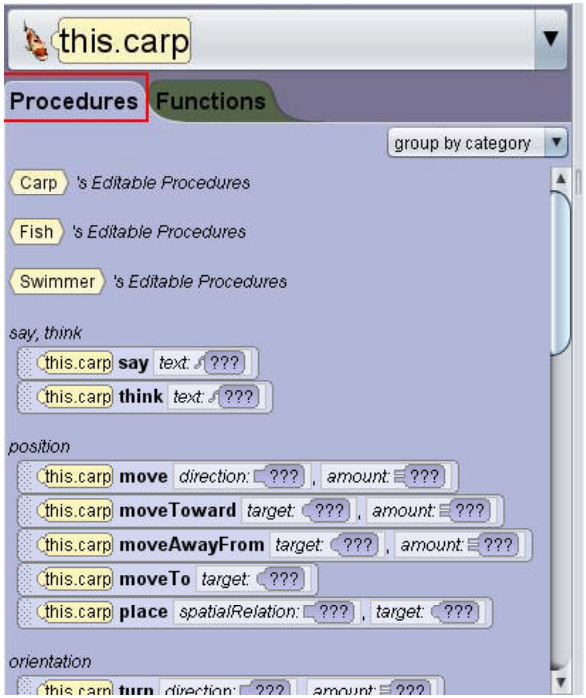
- Procedures: All pre-defined procedures for an object.
- Functions: All pre-defined functions for an object.



Procedures Tab

The Procedures tab displays pre-defined procedures for the selected instance, as well as procedures of your own that you define.

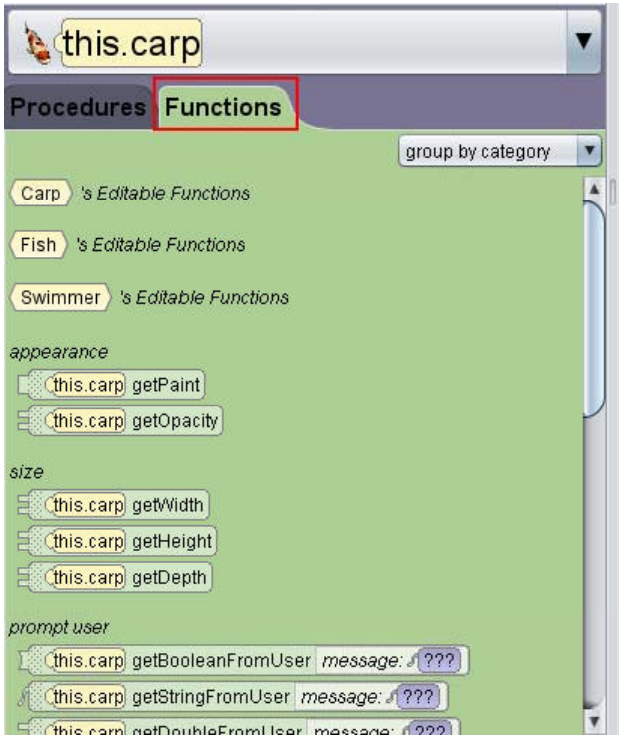
A procedure is a piece of program code that defines how the object should execute a task. Alice 3 has a set of procedures for each class; however, users can create ("declare") new procedures.



Functions Tab

The Functions tab displays pre-defined functions for the selected instance, as well as functions of your own that you define.

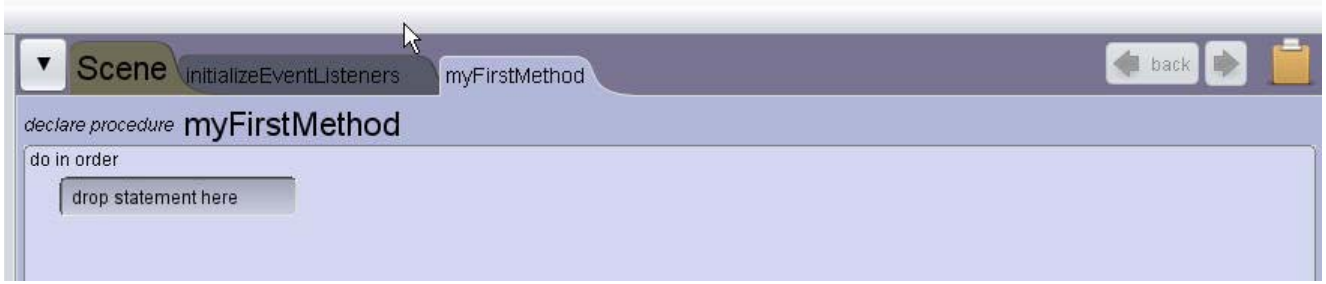
A Function computes and answers a question about an object, such as, “What is its width or height?”, or “What is its distance from another object?” Alice 3 has a set of functions for each class; however, users can declare new functions.



Code Editor Tabs

The Class Menu displays to the left of the Scene tab. You create your programming instructions on the myFirstMethod tab.

By default, Alice creates a Do In Order control statement in the myFirstMethod procedure. The area labeled *drop statement here* is the location onto which you will place programming instructions.



Control Statements

At the bottom of the myFirstMethod tab you will find the Alice 3 control statements.

Object Movement

Object movement is egocentric: Objects move based on the direction they face.

An object can move in six directions:

- Up
- Down
- Forward
- Backward
- Right
- Left

Examples of Movement Procedures

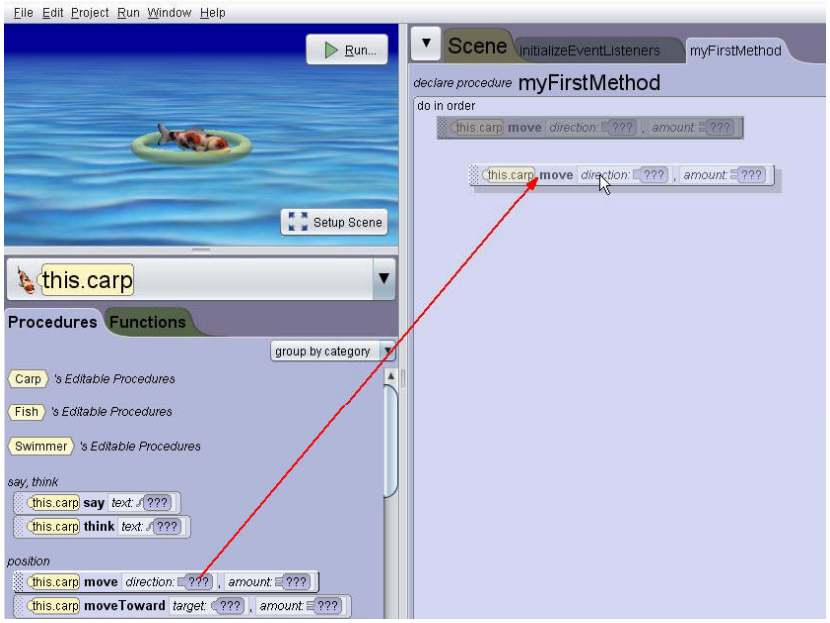
Procedure	Description
Move	Moves the object in any one of its six directions.
Move Toward	Moves the object toward another object.
Move Away From	Moves the object away from another object.
Move To	Moves the object from its current position to the center point of the target object.
Move and Orient To	Moves the object from its current position to the center point of the target object and adjusts the object's orientation to match the orientation of the target object.
Delay	Halts an object's movement for a certain number of seconds. Delay can be used to slow down the execution of an animation.
Say	Creates a call-out bubble with text that makes the object appear to talk.

Examples of Rotation Procedures

Procedure	Description
Turn	Rotates an object left, right, forward, or backward on its center point. Left and right turn on the object's vertical axis; forward and backward turn on the object's horizontal axis.
Roll	Rolls an object left or right on its center point using the object's horizontal axis.

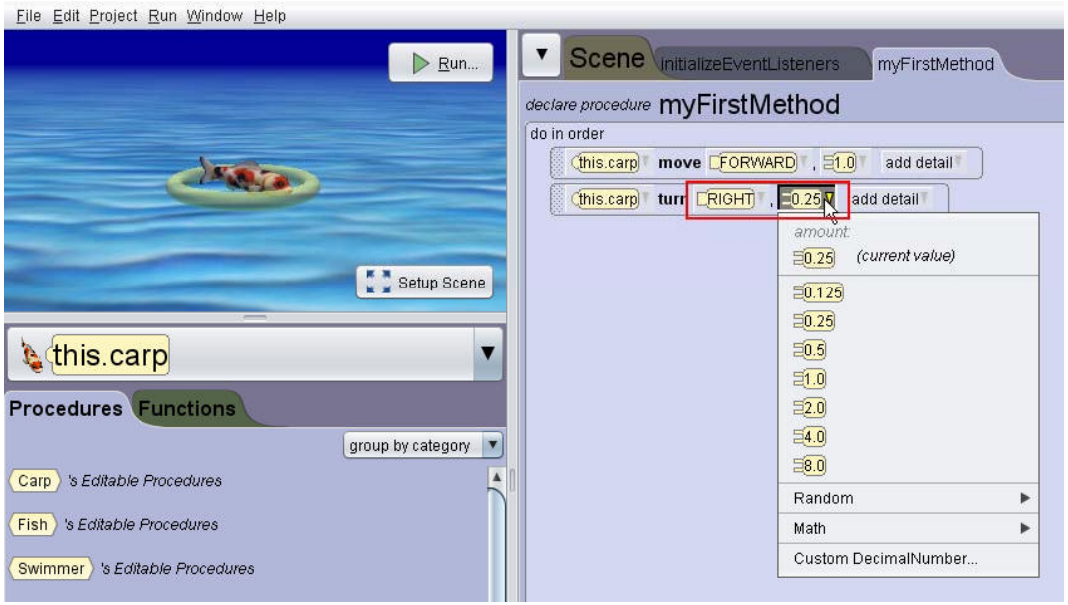
Create a Programming Instruction

From the Procedures tab, click and drag the desired procedure into myFirstMethod in the Code editor.



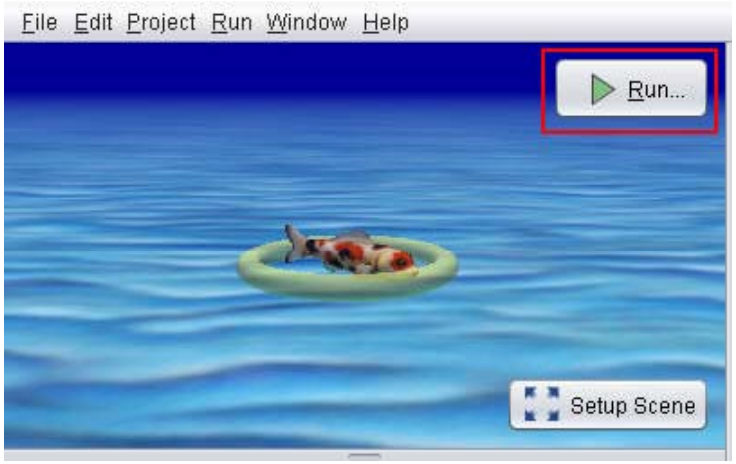
Select and Set Argument Values

After the programming statement is created, use the drop-down menus to set the values for each argument. To access the argument drop-down menu, click on the down-pointing triangle to the right of the argument value.



Execute the Program

Click the Run button to execute the programming instructions. Run the program frequently to test for the desired results, and alter the values of the arguments as necessary.



Arguments

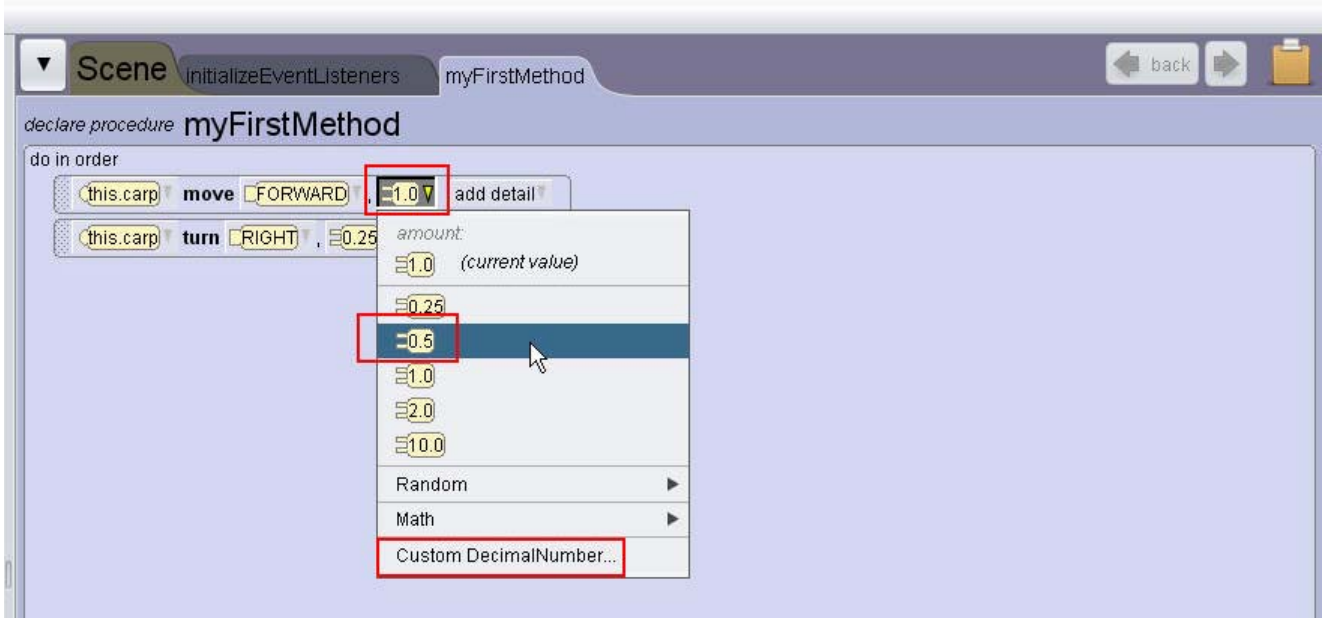
Arguments are selected after the procedure is dropped onto the Code editor. Argument types may include:

- Object
- Direction
- Direction amount
- Time duration

An argument is a value that the procedure uses to complete its task. A computer program uses arguments to tell it how to implement the procedure.

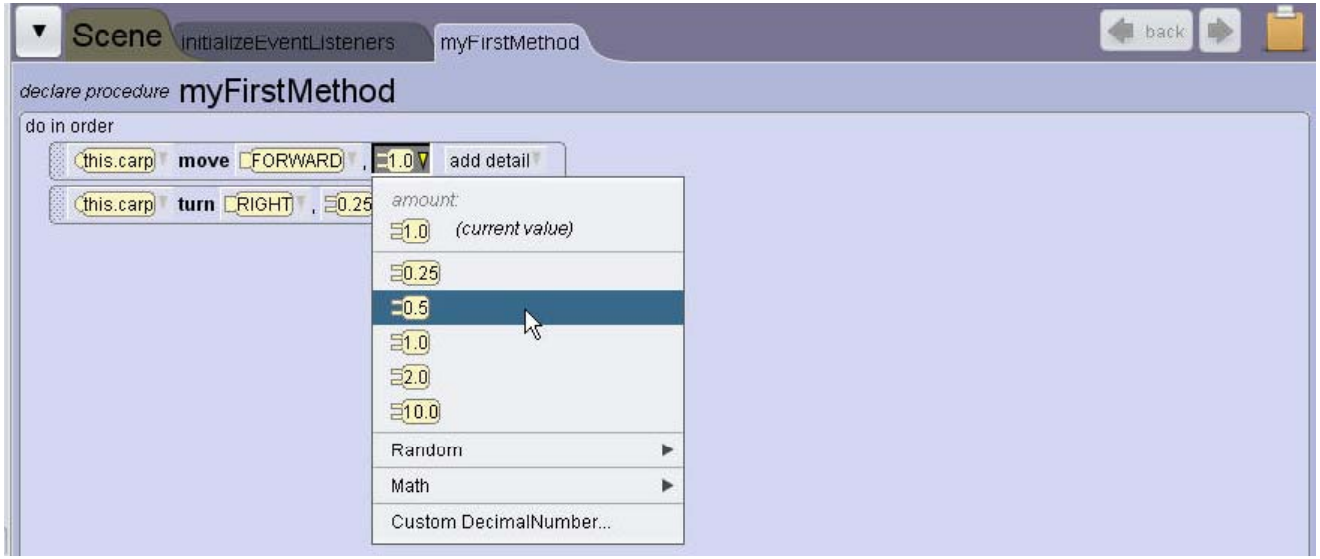
Argument Menu

The argument menu offers default argument values to select from. If none of the defaults are suitable, select the Custom DecimalNumber… option so that you can specify a more accurate argument value.



Steps to Edit Arguments

1. Next to the argument's value, click the arrow to display the menu of values.
2. Select a new value.
3. Use the Custom DecimalNumber… option to specify a value that differs from the default list of values.



Arguments as Placeholders

When a procedure is dropped into the Code editor, all argument values must be specified. There will be times that you select an argument value as a temporary placeholder value that will be replaced later.

For example, you may want an object to move forward but you are not sure how far. Select a placeholder value of 2 meters, run the animation, determine that a different value is needed, and then edit the value. You can also specify a placeholder value that you will replace later with a function or a variable.

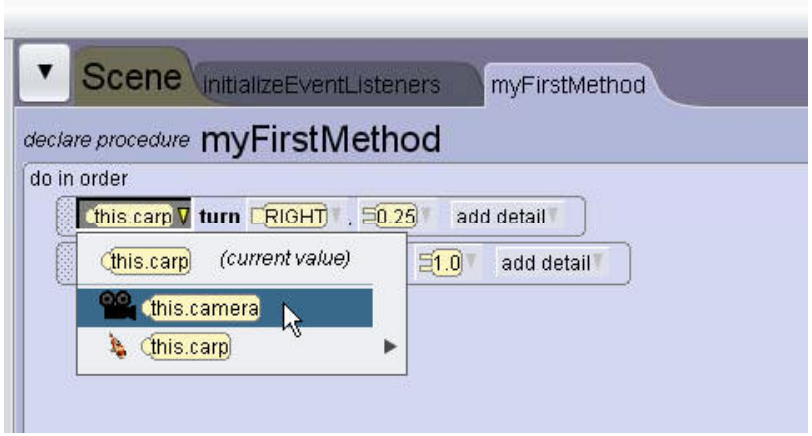
Steps to Reorder Programming Statements

1. Use “Drag and Drop”: Select the handle of the programming statement.
2. Drag the programming statement to its new position.
3. Drop the programming statement into its new position by de-selecting the handle.*

*Note: A green position indicator line will appear to help you align the programming statement to the desired position.

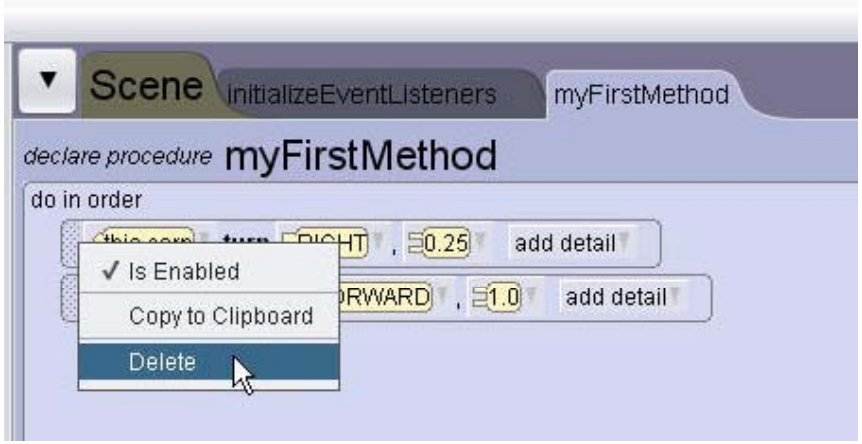
Edit Programming Statements

Use the drop-down lists to edit the values of a programming statement.



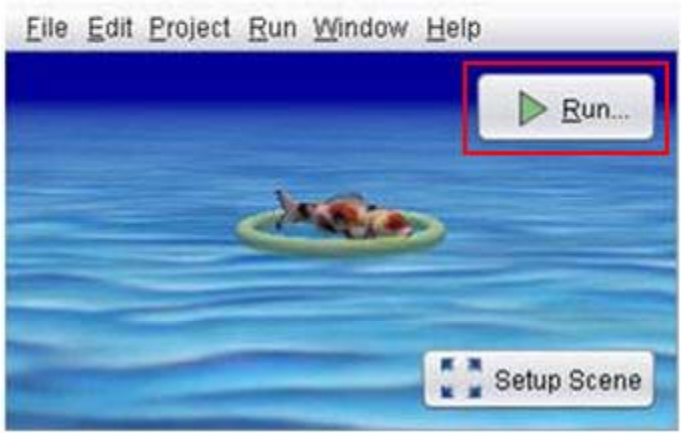
Delete Programming Statements

Right-click programming statements to delete them.



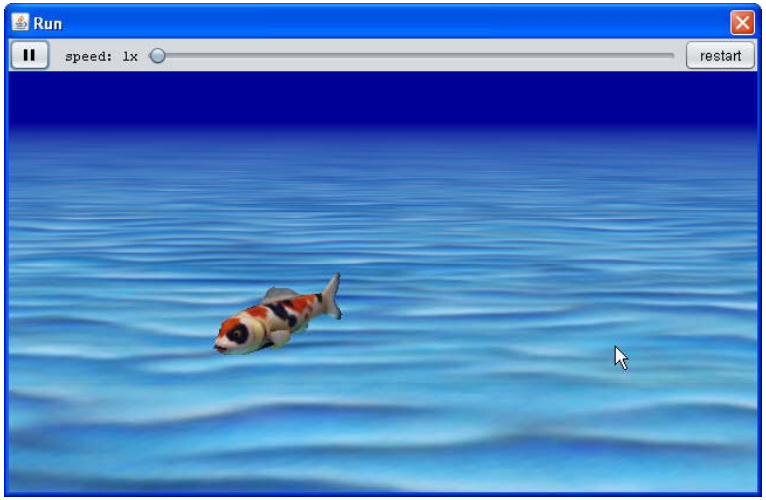
Edit and Test the Program

Run the animation to test it, and edit the code as necessary. It may take several cycles of testing and editing to get the animation to run as desired.



Debugging and Testing

Debugging and Testing is the process of running the animation many times, and adjusting the control statements, procedures, and arguments after each execution. Save often while debugging your program.



Insert Temporary Programming Statements to Help with Debugging

You can insert temporary programming statements into your code to help with debugging. For example: imagine that you have an object that is not moving forward.

- Temporarily enter a Say programming statement to announce that the object is about to move forward.
- Test the program to see whether or not the Say programming statement executes.
- If the Say statement executes, but the object does not move, this indicates one type of problem; if both the Say and Move statements do not execute, this may indicate another type of problem.

Disabling Programming Statements

Programming statements can be disabled in the Code editor.

Disable programming statements to:

- Help isolate portions of code during testing.
- Help you focus on programming, testing, and debugging a specific instruction.

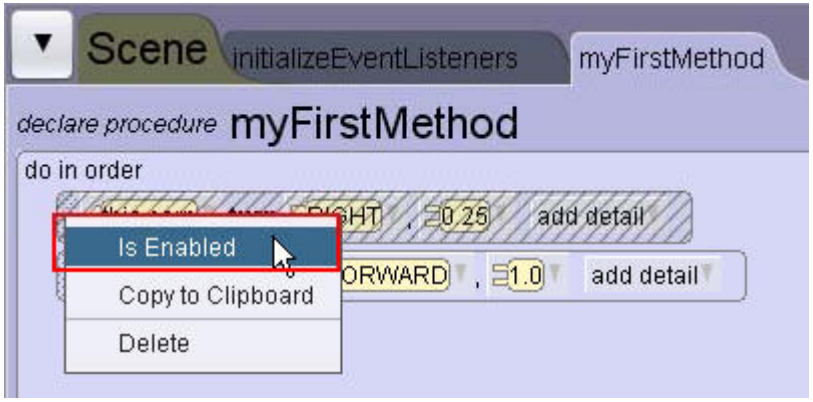
Steps to Disable a Programming Statement

1. Right-click on the programming statement.
2. Unselect "Is Enabled" from the drop-down list.

3. The color of the programming statement will change to gray hashmarks to show that it is disabled.

Steps to Re-Enable a Disabled Programming Statement

1. Right-click on the programming statement that has been disabled.
2. Select "Is Enabled" from the drop-down list.



Programming Comments

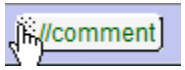
Including programming comments in an animation helps humans understand the flow of the programming.

Comments:

- Describe the intention of the programming instructions.
- Do not affect the functionality of the program because they are ignored during its execution.
- Are typically placed above the block of programming statements that it describes.
- Are often written first, in large programs, as an outline of the programming instructions.

Steps to Enter Comments in a Program

1. Drag and drop the comments tile above a code segment. The comment tile is located at the bottom of the myFirstMethod.



2. Type comments that describe the sequence of actions in the code segment.

Using Comments to Organize Your Program

Comments can be an excellent tool for organizing the development of a program.

For large programs, create a comment that indicates the end of the program. An “end of program” comment helps to minimize scrolling when adding programming statements to a lengthy myFirstMethod.

Summary

In this lesson, you should have learned how to:

- Toggle between, and describe the visual differences between, the Scene editor and the Code editor
- Locate and describe the purpose of the methods panel and the procedures tab
- Use procedures to move objects
- Add Java programming procedures to the Code editor

Summary (cont.)

In this lesson, you should have learned how to:

- Demonstrate how procedure values can be altered
- Create programming comments
- Reorder, edit, delete, copy, and disable programming statements
- Test and debug an animation