Alice Facilitation Guide

Programming in Alice

(https://www.alice.org/resources/lessons/programming-in-alice/)

Summary

This facilitator guide is intended to guide the instructor through introducing the use of the Alice Code Editor for creating the program, or script for a story or game.   This guide is intended to guide the facilitator through the introduction of the Alice built-in procedures, the fundamental building blocks of Alice programs. This includes a basic overview of the code editor interface and some basic code building skills, provide options for participants to use the code editing tools, and debriefing the experience at the end. This should be one of the early lessons in The Alice Project's curriculum to learn how to create an animation or interactive project. This lesson is intended to be the minimum lesson for being able to program in Alice with other lessons expanding on different aspects of the code editor that can be mixed and matched to build upon this lesson as desired. It is assumed that this lesson would follow The Alice Scene Editor lesson or some variation so that a basic understanding of object orientation and how to add and remove objects has been taught. It can be offered as a stand-alone lesson in basic programming in Alice where the product of this lesson does not carry forward to the next lessons using a broad range of supplied exercises and projects around using the code editor. It can also be used to kick of a longer project based curriculum where the program built during this lesson is later used as a starting point for building upon in later lessons.

Learning Objectives

- What makes up an Alice program
- Code Editor Overview
- What is a Method
- What is an Alice statement
- What is a Parameter and Argument
- How to add a procedure to an Alice program
- How to run an Alice program

- How to add a control structure to your program
- How to edit your program
- Where to find additional resources for the code editor and using procedures

Lesson Overview

- Introduction
- Lesson on the Alice Code Editor
- Student Work Session
- OR Step-by-Step Work Session
- OR Guided Student Work Session
- Assessment
- Debrief / Students Share Work

Skills Overview

This project was developed for use with Alice 3. The following Alice 3 skills will be learned through the lesson and additional resources for them are linked to in the How to Resources section and in the tutorial exercise. Optional educational activities can be incorporated based on their relevance to the required steps for the project.

Alice Basics

Navigating the Alice Code Editor

Adding procedures to an Alice program

Running an Alice Program

Editing and Modifying Statements

Modifying a procedure's execution with add details…

Changing a procedure's position in myFirstMethod

Modifying a procedure's arguments

Using a procedure on a subjoint

Copying and Pasting a Procedure

Deleting a procedure

Control Structures

Using a Do in Order and a Do Together

Camera Markers

Using Camera Markers to Program Camera Moves

Prep + Materials

Classroom Resources

Computer Access

Each participant should have his or her own computer for the duration of the project. It is also possible to allow pairs of students to work together at a shared computer.

Presentation + Lecturing

Ideally, you should be able to present the lesson ppt in front of the class. Depending on your approach, you may also want to be able to show Alice and be able to demonstrate and guide the class through the exercise. You can also print and distribute these materials if needed.

Supporting Materials

You may want students to have access to the How To… resources that accompany this lesson and are linked to on the webpage associated with this guide. This can be achieved by insuring they have access to the Alice.org website and can play the videos or by downloading and making the videos accessible in another way. You may also wish to download, print, and distribute the accompanying How To... quick reference guides that can be found on the relevant How to webpage. These materials can be downloaded in .doc format to combine several into one hand out.

Software Requirements

This lesson requires each computer to have Alice 3 installed and available.

Time

The lesson is designed to take 45m-90m, depending on:

- The inclusion of the lesson presentation
- Time spent on going over skills training
- Time spent on optional learning activities
- Complexity of exercises or projects assigned
- Time spent debriefing

Suggested Process

Introduction

Tell the students that they will be shown how to use built-in procedures to create a simple program in the Alice development environment. Describe the class activity and discuss the skills they will acquire in the process.

Computer Program and Alice Code Editor Lesson (optional)

For this lesson, you can present the lesson materials in a couple different ways due to the inclusion of a lot of Alice skills components in this lesson.

Option 1

Run completely through the supplied power point slides to give an overview of the editor, the concepts, and the skills lessons. More detailed talking points are provided below for highlighting key concepts while giving the presentation.

Option 2

Integrate a simple "hello, world" exercise to break up the lecture and give the students a chance to write their first program. This can be interjected before moving on to control structures immediately following the directions on how to add procedure.

Option 3

Integrate application demos or the more in depth How To… videos into the presentation at the relevant topic moments. The level of detail and time spent on the skills details may be determined by how you plan to integrate the exercise component (see below), accessibility of the supporting materials to the whole class, and the skill level of your students.

Exercise Facilitation

There are several options for how to allow the participants to explore the code editor and apply the new skills.

Option 1 (Open Student Work Session)

With groups that do well independently you may choose to have them build a scene of their own design, assign them one (or several) of the provided challenge exercises (linked on website page associated with this lesson), or allow them to select an exercise or project from the list provided. Note that the guided tutorial exercise (Building a Program Tutorial) provides the most scaffolding for students that may need that level of support and direction. Additionally, you might want to provide them with access to the scene editor How To… materials to support them.

Option 2 (Guided Work Session – Directions)

For more structure, you may decide you want to assign the students the tutorial exercise for this lesson. The tutorial exercise provides step-by-step directions for guiding the participants through building a scene that points them to the correlating How To… materials at the appropriate points. This exercise also guarantees that they will explore all the different methods that can be used for each skill in the process of completing the exercise. You can point the students to the web page for the exercise or print out the associated directions.

Option 3 (Guided Work Session – Instructor Led)

For the most structured and supported format, you may break the session up into smaller segments or Modules. Each module includes demonstrating more in depth each skill before having the participants apply them through the activity. This format also provides more break points to check in with participants. Detailed step-by-step guide provided below. You can approach this in two ways

- Play the more general How To… videos or demonstrate the skills and then allow participants to then follow the step-by-step directions for the module,
- Or demonstrate the exact steps from the exercise to the class and then have them do the step demonstrated.

A guided facilitation guide is provided below with more details. The session would follow this basic flow:

Module 1: Getting Started

Step 1-5. Get everyone to open Alice, create or open the described scene, navigate to the Code Editor.

Module 2: Adding Procedural Methods (Statements)

Step 6-8. Add statements to myFirstMethod for the appropriate objects from the Procedures tab of the Methods panel. Save the program. Ensure everyone has successfully added procedures to myFirstMethod in the Editor.

Module 3: Testing, Editing, and Modifying Procedures

Step 9-13. Run the program. Modify existing parameters. Add new optional parameters through add details to modify the program. Ensure everyone has successfully modified procedures in myFirstMethod in the Editor.

Module 4: Programming SubJoints

Step 14-17. Use procedural methods to animate subjoints of an object. Ensure everyone has successfully added subjoint procedures in myFirstMethod in the Editor.

Module 5: Using A Do Together

Step 18-19. Add and nest Do Together and Do in Order control structures to myFirstMethod to modify the program. Ensure everyone has successfully added an assortment nested Do Together and Do in Orders in myFirstMethod in the Editor.

Module 6: Programming a Camera Move

Steps 20-22. Add camera markers in the scene editor and use procedural methods to animate a camera move. Ensure everyone has successfully added a camera move in myFirstMethod.

Assessment (Optional)

You can use the supplied bank of assessment questions, challenges, and exercises to quiz your students on the retention of their new skills. These materials are provided in a separate document that can be downloaded from the webpage associated with this guide. A word document has been provided to allow you to customize as needed.

Class Regroup + Summary

We recommend regrouping as a class to discuss challenges and successes, and to offer feedback, both among the participants and about the curriculum itself.   There are provided reflection questions found below.

Standards and Integration

Interim 2016 CSTA K-12 CS Standards

Algorithms and Programs - 1B-A-2-1:
Apply collaboration strategies to support problem solving within the design cycle of a program

Algorithms and Programs - 2-A-2-1:
Solicit and integrate peer feedback as appropriate to develop or refine a program

K-12 Framework Integration

Lesson Material Talking Points

The Alice Virtual Environment
What is a virtual environment? Basic elements of an Alice scene.

Slide 3 What is a Computer Program?

- A program is a list of instructions for a computer to perform a task or solve a problem
- Programmers generally write computer programs using programming languages which can then be translated into machine code that can be executed by the computer

Slide 4 Basic Programming Ideas

- A program can be a list of instructions to follow in order (sequential processing)
- A program can be a direction to do something if something else happens (conditional execution)
- A program can be instructions to repeat a behavior a certain amount of times or until something happens (looping or iteration)
- A program can be a a set of directions that returns an answer to a question (function)
- A program can be a mixture of all of the above in differing amounts and combinations

Slide 6 Programming with Alice

- Alice is a programming tool designed to introduce students to fundamental programming concepts
- It would not be used to solve every type of computing problem or task
- The focus of Alice is to develop animations; stories, games, simulations
- Alice is an object-oriented programming environment.
    - In object-oriented programming, computer programs are created by building classes with properties and behaviors, called methods.
    - These methods can be thought of as directions to be followed.
- All computer programs, including Alice animations should start with a design
    - See the Lessons on design and algorithm development

Slide 7 An Alice Program

- different required elements of an Alice Program
- Directions for how the computer will build and display the environment – created using the scene builder
- Directions for what the happens in the world and when –created using the code editor
- Directions for what the program should listen to from outside the program (key inputs, mouse interactions) – programmed using the code editor

The Code Editor
How do you program in Alice? What are the basic elements in the Alice code editor?

Slide 8 Scene Editor Overview

The Scene Editor has three main panels

- The **Camera View** will display the Scene as it is being built. It contains the camera controls and camera view menu to allow different perspectives of the Scene as it is being built. The objects you add to the scene can be manipulated directly in this window.
- The **Gallery** contains collections of 3D models that can be used to build a scene.
- The **Properties Panel** provides buttons for changing settings for the camera window, selecting objects and changing their positioning, size, color, and other properties, and for adding and manipulating unique camera and object markers

What is an Object?

Classes and Objects. How Alice uses the Gallery to organize the classes that can be used in a virtual environment.

Slide 9 Code Editor Overview

- Writing an Alice program takes place in the Code Editor, and involves using, modifying, and creating methods for the classes used in the project.
- The Code editor has four different panels… The Camera View, the Editor, the Methods Panel, and the Controls Panel.

Slide 10 Camera View

- The camera view shows the scene in its state at the beginning of the program and from the starting camera view.
- You can find the buttons for accessing the scene editor and for running your program here.

Slide 11 Methods Panel

- The Methods panel contains tiles or directions that are placed in the Editor for the creation of the program code.
- In the methods panel has a drop down to select the object you want to work with and tabs to access either procedural or functional tiles.

Slide 12 -13 The Editor

- The Editor panel of the Alice environment is where code creation takes place.
- The editor panel can be navigated by clicking on the tabs along the top of the editor as well as the class button drop down menu to the left of the tabs.
- There are 2 types of tabs differentiated by the color of the tabs.
  - The yellow tab indicates that you are viewing the information for the class identified in the text of the tab. The class tabs, when clicked on, displays all the methods and properties of the class.
  - The purple tabs in the code editor are method tabs where sets of directions or code can be created.
- Whenever an Alice project is created or opened, Alice always opens the Scene class tab and the myFirstMethod tab of the Scene class.

    - Think of the Scene class as being a stage object and the directions or script for the play being executed on the stage are the methods being called or created for the scene class.
    - The Scene class is where code creation for the project usually starts, in myFirstMethod.
- myFirstMethod is set up to be the method or set of directions that is first executed when the Run button is clicked.

Slide 14 Control Panel

- The control panel contains tools that extend the functionality of the code editor.
    - The commenting tool allows you to add notes to you or others that will be ignored by the computer
    - There are different control structures to allow you to structure your code to execute things such as loops and do together
    - There are tools to allow you to add and manipulate variables in your program
- Don't worry these things will be covered in more detail in later lessons.

An Alice Method

What is a method? How do they relate to Objects in a Scene?

Slide 16 What is a Method

- In Alice the term method is used to describe a small program or set of code that can be called by the program to do something.

Slide 17 Types of Methods Procedural/Functional Methods

- The Methods panel uses two tabs to display two different types of methods:
    - Procedures (procedural methods), which are methods or directions that perform an action
    - Functions (functional methods), which are methods that compute a value and return an answer or response.

Slide 18 Fundamental Alice Procedures

- There are built in fundamental procedures to enable the basics of animation. Many of the Alice procedural methods are some variation or combination of move, turn, and roll.
- move changes the location of the object in the scene, and by itself will not change the orientation of the object
- turn and roll change the orientation of the object in the scene, and by itself, will not change the location of the object
- There are also some other built in procedures to allow manipulation of the environmental effects of the scene such as lighting, fog effects, audio, and visibility of objects.

Slide 19 Object Oriented

- It is important to remember that Alice is an object oriented development environment so when you want to write your program you do so by navigating to the object you want to control to access the available procedures and functions.
    - Different objects will have different available procedures such as the scene object having the atmosphere procedures and the camera having object move procedures

An Alice Code Statement

Program statements in Alice have required fields that are called parameters. What are parameters? What is an argument?

Slide 21 What is a parameter?

- When a procedure is added to the code editor a series of sub-menus that represent parameters for that procedure.
- Different procedures will have different parameters. For example:
    - move will ask for direction and distance in meters to move
    - turn will also ask for direction, but will ask for amount (in percentage) of the rotation
    - Without having information for these parameters the program won't know what to do

Slide 22 Required parameters

- Alice won't let you add incomplete statements to insure that your code is always complete and functional. To do this there are certain parameters that Alice requires you to give arguments or values for to be able to add the procedure to your program.
    - For example you can't add the move statement to your program without giving a value for the direction and distance
    - Default values have been added to the drop down but in most cases, there will also be a custom input option that allows you to type or use a number input to choose your own value.
    - These argument values can be modified later so don't worry too much when first selecting.

***Slide 23* Optional Parameters (Add Details)**

- Almost every Alice procedure has an *add details…* or parameters associated with it. These arguments are optional.
- These details modify the way the procedure behaves in some way.
- The most common details
    - **duration:** specifies how long the animation will take (default value is one second)
    - **style:** specifies how the animation executes
    - **as seen by**: specifies the point-of-view the animation will use to execute

Adding Procedures to the Editor
Program statements in Alice are created by selecting the object that will execute the procedure, dragging the procedure tile into the Editor, and selecting the appropriate arguments for the procedure.

Slide 25-26 Selecting Object, Drag and Drop, Fill in Required Fields

- The Objects Menu, located underneath the Camera View and above the Methods Panel, displays all the objects currently available to the programmer when it is clicked on.
    - It should be noted that the keyword this in this menu refers to the Scene object.
- To add procedures for the subjoints of an object, select the object's SubJoint menu, (assuming the object has subjoints), by clicking on the right-arrow triangle in the menu.
- Click on the Procedures tab, if it is not already selected.
    - You will notice that the subjoints have a different set of procedures available compared to an object.
- Click and drag the selected procedure tile into the Editor
- A green line will appear in the Editor, indicating where the new statement will appear
- When there are already program statements in the Editor, it is possible to insert the new statement before or after those that are already there.
- When a procedure is added to the code editor a series of sub-menus will appear depending on the arguments required to execute the statement.
    - An argument is information that a statement needs to execute the instruction
    - For example, Alice will prompt for the direction and the distance for an object's move.

Running Your Program
Alice allows you to Run your program as often as you want. This is a great way to test and iterate your work.

Slide 28 The Runtime Window

- Pressing the Run button found on the camera view panel will launch the runtime window and auto play your program
    - The window can be resized by dragging the corner or entering full screen
    - The player controls allow you to pause, change speed, or restart your program
- Right Clicking on a statement in your program and selecting Fast Forward will open the window and rapidly execute the animation to that statement in the program, and then the animation will execute at normal speed from that point on
    - Also useful for testing procedures and debugging the program

Slide 29 Runtime Errors

- Even with all of the built in checks to try to insure only complete code is created there are ways a program or Alice can fail
- The error dialogue will hopefully help you understand what is wrong with your program and the application
- In some cases you can click through the error and in others you will need to go back and fix your program before being able to play

- In some cases you may just need to restart Alice. It is a good idea to save your world before running your program just in case a fatal flaw is encountered

Using Control Structures
To create more complex programs you will use elements from the control panel. The two most common control structures you will use are the Do in Order and the Do Together

Slide 31 Do In Order
- Do in Order is the default behavior for method tabs in the code editor
- Do in Order treats your statements sequentially so order is very important

Slide 32 Using Do Together
- The Do Together is one of the control structures found in the control panel at the bottom of the Editor window
- The Do Together tells the computer to sequence the directions inside the block at the same time
- Be aware that if you put two opposite commands inside a do together they will happen simultaneous and you may see no effect in your program. Ie if you move a joint forward and backward at the same time what do you think you will see?

Slide 33 Nesting Control Structures
- Alice allows you to put control structures inside of other control structures. This level of complexity may be needed to achieve the behavior you desire.
- The example shows how you might structure two objects jumping at the same time. The nested Do in Order is required to allow the program to execute the move while at the same time having them both happen at the same time.

Editing Your Program
Once statements have been added to the program in the editor they may be modified by changing the statement arguments, moving the statements to different positions in the Editor, deleting and disabling statements.

Slide 35 Undo/Redo
- There is no Undo / Redo button as there is in the Scene Editor
- Can be found in the Edit menu
    - The Cut, Copy, and Paste commands shown in the Edit menu are not implemented at this time
- Keyboard command in the Code Editor
    - Windows: Control+Z, Control+Y
    - Mac OS X: Command+Z, Command+Y

Slide 36 Changing Arguments
- Code statements may be modified in the editor, by clicking on the yellow argument components in the statement, (these will have a small arrow next to them).

- Selecting these arguments will bring up a drop-down menu showing the available options for substitution.
- It is even possible to change the object that is performing the instruction

Slide 37 Changing the Order of Statements
- Statements may be re-ordered by dragging them from position to another within the Editor
- The green line will appear, indicating the new position insertion point

Slide 38 Deleting Statements
- An instruction may be deleted from the editor by dragging the instruction tile to the methods panel
- An image of a trash can should appear
- Or an instruction may be deleted by right-clicking on the instruction and choosing delete from the menu that appears.

Slide 39 Copying and Duplicating Statements
- A statement may be copied to the clipboard, and then the clipboard can be dragged to a new insertion point in the program, placing the copied statement there
- You can option click on mac or option click on pc to duplicate a statement or block and add it to another location
    - This can be very helpful for creating similar statements that you can then change arguments to replicate them for another object or create incremental or oscillating move commands
    - You can copy individual statements or whole nested control structures

Slide 40 Disabling Statements
- You can disable and enable code through the right click drop down.
    - A disabled statement will be ignored when the program runs.
    - This is a good way to focus on testing specific procedures when debugging your program

Tips and Tricks
Some simple tips and tricks to remember when you are using the code editor.

Slide 42 Notes About Other Alice Procedures
- moveTo:
    - moving one object to the location of another object so that both pivot points are at the same place
    - The objects will look like they are overlapping; Alice does not have built-in collision detection
    - The orientation of the moving object does not change
- orientTo:
    - Object does not move

    - The objects orientation will change to be in alignment with the targets orientation
    - moveAndOrientTo:
    - moving one object to the location of another object so that both pivot points are at the same place
    - The orientation of the moving object will change to the same orientation of the target object
- turnToFace:
    - Turning an object around its pivot point, so that its forward orientation will be in the direction of the target
    - Does not change location
- pointAt:
    - Turning an object around its pivot point, so that its forward orientation will be in the direction of the target's pivot point
    - Does not change location
- say / think:
    - uses TextStrings (sequences of letters and digits) as arguments to create speech or thought bubbles
    - optional detail arguments will allow modification of the speech / thought bubbles
    - Programmer cannot modify placement of the bubbles. Alice makes its best guess as to where the bubbles will appear.

Slides 43 Animating the Camera
- Camera moves can be a very important part of viewing, animating, or moving through a scene.
- You can animate the camera the same way you can animate any other object in Alice
- You can also use the unique camera marker object to plan and program camera moves to specific positions

Slide 44 Scene Procedures
- The scene object has several unique procedures that allow you to manipulate scene properties such as lighting and fog
- These can be programmed to create very cool visual effects

Slide 45 Incremental Development
- Encourage the students to develop an incremental development style
- Write a little code, test what is written by running it.
- It is much easier to find and correct mistakes while developing the program

Slide 46 Save Often
- Emphasize the importance of frequently saving projects, using a versioning naming system

- Projects can become corrupt while students are working on them, for a wide variety of factors. Students will not have to restart a program from the beginning if they have earlier versions of the project

Exercise Facilitation Step-by-Step

These step-by-step directions are for the guided facilitation option 3 that uses the Scene Editor Tutorial as a basis for the hands-on experience for the session. They can be followed in addition to having first gone through the whole ppt lesson.

Module 1: Setting up the Scene

Goal – Complete Steps 1-5 of Tutorial Exercise

Students will be able to open Alice, load a starter world or review building a basic scene, become familiar with the code editor.

Media

- Play the video: Scene Editor Overview
- OR Demonstrate selecting and opening a saved world

Talking Points

- Remember that you can use the Scene Editor How to Videos to refresh the use of the scene editor
- When placing an object off-stage you can place the object where you want it to end after it's movement and use one shots to move it a specified distance and direction that you can keep track of and then use to program the entrance
- Remember that if you can't find an object after placing it that it may be obscured or inside of another object

Module 2: Adding Procedures

Goal – Complete Steps 6-8 of Tutorial Exercise

Students will be able to add procedures with appropriate arguments to myFirstMethod and be able to use the basic move and say procedures.

Media

- Play the relevant parts of the video: Procedures Overview
- OR Demonstrate adding a procedure to the Editor

Talking Points

- Methods (procedures and functions) are part of the classes that are used in the animation. Alice objects come with a set of built-in procedures. Other lessons will show how the programmer can create new procedures and functions for an Alice class.
- The Object Menu, underneath the Camera View in the Code Editor, is the list of all the objects that are currently in the Scene. Select the Object that will execute the procedure.
    - It should be noted that the keyword this in this menu refers to the Scene object.
- Make sure that the students have selected the Procedures tab in the Methods Panel.
    - You might also check to be sure that the student has selected the myFirstMethod tab in the Editor.
- Drag and drop the selected procedure into the Editor. A green line will show where the procedure statement will be placed. It is possible to insert the procedure anywhere in the Editor.
- Almost every Alice procedure requires information – arguments – to successfully execute.
    - Alice tries not to permit incomplete statements in the code, and so drop down menus will appear, asking that the information be provided. If values are not selected from the menus, then the statement will not be added to the Editor.
    - For example, a move statement needs to know which direction to move, and the distance to move. A turn statement needs to know which direction to turn, and the amount of the rotation.
    - Even if you are not sure of the final value to be used, you must select a value, which can be modified later

Module 3: Testing Editing and Modifying Your Program

Goal - Complete Steps 9-13 of Tutorial Exercise

Students will be able to run their programs and modify the behavior of the procedures, editing required arguments and adding optional arguments.

Media

- Play the relevant parts of the video: Procedures Overview
- OR Demonstrate the various techniques for modifying and editing procedures in the Editor

Talking Points

- Modifying the arguments of the program statement in the editor
    - By clicking on the arguments in a program statement, a drop-down meu will appear that will allow the selection of appropriate argument values, including changing the object executing the statement.
- The add details… button on almost every procedure provides a menu of additional arguments for the program statement.
    - The most common are
    - duration: determining how long it will take the animation to execute. By default, all Alice animations will execute in one second.
    - animation style: determines how an animation will start and stop when it executes
    - as seen by: determines the point of view (or orientation) that wil be used by the object when it executes
- Inserting / repositioning statements in Editor
    - A green line indicates where in the Editor the procedure tile will be placed when dragged from the Methods panel
    - It is also possible to move a statement that is already in the editor to another position in the program code by clicking and dragging the tile into a new position, also indicated by the green line
- Deleting statements from the editor
    - A statement may be deleted from the program code in the Editor by clicking and dragging it back to the Methods panel. A Trash Can icon will appear in the Methods Panel to indicate that the statement will be deleted
    - It is also possible to right-click on the statement tile in the Editor and from the context menu that appears, select Delete, which will remove the tile.

Module 4: Programming Subjoints

Goal - Complete Steps 18-19 of Tutorial Exercise

Students will be able to select subJoints from the object menu and add procedure statements for subJoints.

Media

- Play the video: Using a Do Together
- OR Demonstrate setting up a Do Together

Talking Points

- The Control Panel, located at the bottom of the Editor, displays all the control structures available in Alice.
- To add control structures to the Editor drag them into the Editor similar to a procedure tile. You can then add or drag existing statements into the block to build.
- You can add additional control structures inside of other control structures.

Module 5: Programming a Camera Move

Goal - Complete Steps 20-22 of Tutorial Exercise

Students will be able to add camera markers and use them to program camera moves into their animations.

Media

- Play the video: Setting Up and Using Camera Markers
- OR Demonstrate setting up a camera marker and creating a camera move procedure

Talking Points

- You can animate the camera the same way you can animate any other object in Alice.
- You can also use unique camera markers to help program camera moves.
- Camera markers are set up in the scene editor and have both a location and an orientation.
- You can use Camera markers to move and orient the camera object by using them as a target for camera procedures.
- You can add additional control structures inside of other control structures.