Alice Facilitation Guide

Control Structures Overview

(http://www.alice.org/resources/lessons/control-structures-overview/)

Summary

This facilitator guide is intended to guide the instructor through introducing the programming concept of control structures and give an overview of the different types of control structures provided in Alice.   This guide is intended to guide the facilitator through the introduction of the different control structures, give an overview of how they are constructed, and give examples of some of their different uses within an Alice program. This includes a short presentation on control structures, provides some options for exercises to make use of the basic do in order and do together control structures, and a guide for debriefing the experience at the end. This lesson is intended to be an overview to control structures and does not go into details on how to setup and use each of the different structures in Alice. There are separate lessons for each of the major control structures in Alice that go into more depth and have different prerequisite and knowledge. As such this lesson can be integrated into the introduction to programming in Alice lesson to extend the section on do together and do in orders, or you can use it later as a stand alone overview to better introduce the concepts and the use of do together and nested blocks, or it can be paired with or immediately followed by one or more of the more in depth lesson for each of the different control structures. Some of the concept covered also make use of functions, variables, and events. If you haven't already covered these concepts you may decide to include them with this lesson or pair them with the more in depth control structure lessons. The more in depth lessons have direct prerequisites outlined for which lessons will require what knowledge in order to fully implement based on the following possible roadmap or level of requirements:

Control Structures Overview Do Together
Prerequisites: none

Count loop
Prerequisites: none

Each in Together

Prerequisite: Arrays and Variables

For Each In

Prerequisite: loops, Arrays, and Variables

If else

Prerequisite: Functions for Basic - Variables for Advanced

While

Prerequisite: loops, and variables or functions

Learning Objectives

- Understand the concept of a control structure
- Learn the four types of control structures in Alice
- Understand what control structure in Alice can be used for sequential execution
- Understand what control structure in Alice can be used for simultaneous (concurrent) execution
- Understand what control structure in Alice can be used for conditional execution
- Understand what control structure in Alice can be used for repetitive executions
- Understand how and why to use nested code blocks
- Learn how to synchronize and smooth out Alice animation when using control structures

Lesson Overview

- Introduction
- Lesson on the Alice Scene 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 of the associated webpage for this guide and also linked to in the tutorial exercise associated with this guide. Optional educational activities can be incorporated based on their relevance to the required steps for the project.

Alice Control Structures
Basic understanding of the different types of control structures
How to convert or dissolve control structure blocks

Animation Techniques
Using Do Togethers for complex movements
Adjusting duration parameters to synchronize animations

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 introduced to control structures and shown how to use *do in order* and *do together* to create more complex and interesting animations in the Alice development environment. Describe the class activity and discuss the skills they will acquire in the process.

Lesson Overview (optional)

For this lesson, you can present the lesson several different ways. One option is to run completely through the supplied power point slides to give an overview of control structures, the concepts, and the skills lessons. You could also integrate application demos into the presentation at the relevant topic moments. The level of detail and time spent on the lesson may be determined by how you plan to facilitate the exercise and the skill level of your students.

Exercise Facilitation
There are several options for how to allow the participants to explore using control sturctures and apply their new understanding and 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 (Scene Building 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 1-3. Get everyone to open Alice, select a Scene template, and navigate to the scene editor.

Module 2: Sequential Execution
Step 6-13. Add several different statements to *myFirstMethod* for the different objects from the ***Procedures*** tab of the **Methods** panel. Choose the appropriate arguments for the selected procedures. Save the program and predict the actions of the animation before **Running** the program. Move the program statements around in the Editor and predict how these changes affect the animation. **Run** the program to test the prediction. You may have students look at other student programs and repeat the predict and test process. Ensure everyone has successfully executed several different versions of their program.

Module 3: Simultaneous Execution
Step 14-22. Add a do together code block to the program. Add more statements to the program if desired. Ask the students to move different combinations of statements into the do together and predict. Ensure everyone has successfully added the do together in myFirstMethod in the Editor.

Module 4: Nested Code Blocks
Step 23-28. More complex animations will require the nesting of control structures and code blocks. Ask the students to add a do in order control block to a do together code block. Ask the students to move different combinations of statements into the do in order and predict the results. Students will be made aware of important tips and other features of the Alice 3 Code Editor interface. Ensure everyone has successfully nested control structures in myFirstMethod in the Editor.

Module 5: Synchronization

Steps 29-30. The use of control structures in the creation of more complex animations may create timing, or synchronization, issues, so that a portion of an animation may lag, or run faster than other elements of the animation. Ask the students to modify the duration values of the code statements to better coordinate the execution of the animation.

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

To Be Added

K-12 Framework Integration

To Be Added

Lesson Material Talking Points

Getting Started
What is a control structure? What are the basic elements in the Alice environment for creating and changing the way that program code runs?

Slide 2 What is a Control Structure?

- A control structure is a programming element that tells the computer which parts of the program to execute and in what order.
- In Alice there are 4 different types of control structures
    - *Sequential - process the contained statements in order*
    - *Concurrent - process the contained statements at the same time*
    - *Conditional* - process the contained statements if the outlined condition is met
    - *Repetition* - process the contained statements repeatedly

Slide 3-4 Alice Control Blocks

- In Alice you add control structures to your program by dragging them in from the control panel in the code editor
- The different control structures will have different required components and formatting for where you can add the statements that will be informed by the directions.

Slide 5 Sequential Execution

- **Do in order** is the default for how Alice will execute statements.

Slide 6 Do Together

- **Do togethers** are very useful in Alice for allowing you to construct your program so that multiple things are happening at the same time
- **Do togethers** are important for building complex animations as well as building projects where multiple objects or states are being managed at the same time

Slide 7 Each in Together

- An **each in together** is a do together code block that also has a built in array that can be used to extend the enclosed statement and have it simultaneously execute the same statement for each of the elements in the array.
- The actual processing of the enclosed statements is a do in order it will just sequence them simultaneous for all elements of the array if implemented.

Slide 8-9 If else and Conditional Constructions

- The **if else** construct is most useful when building interactive worlds or implementing randomization into an animation
- An **if else** statement allows you to have the program check to see if some condition is true or false and execute different paths forward depending on the check.

- **If else** statements are core to building input listeners, managing game states, and is often paired with the use of variables and loops to create more complicated programs.

Slide 10 Looping

- Computers have no problem repeating the same task over and over again. In Alice there are 3 different ways you can tell the computer to repeat itself each with it's own twist on deciding when to stop:
    - **Count** - allows you to set the exact amount of times you want something to occur
    - **For each in** - allows you to have it repeat something enough times to substitute in each of the elements of a list
    - **While -** allows you to have the it repeat continuously until a statement you create becomes true or false

Slide 11 Count Loop

- For more advanced groups you can point out that a numbered loop can be constructed in multiple ways
    - a for each in loop for a numbered list
    - a while loop that has a function that counts up or down until a certain number is true or a conditional math equation such as is greater than 1 becomes true or false.

Slide 12 For Each In

- A **for each in** will repeat the loop for each entry in the associated array or list
- The control block then also allows you to reference that list and have it pass in the current item from the list into your enclosed statements to have it function differently on each repetition

Slide 13 While Loop

- A **while** loop will continue to repeat the contents of the code block until the conditional statement that encloses it changes
- You can set up an infinite loop by leaving the loop control block set up with while true is true
- To set up a while loop that can be escaped you will need to use a variable or math function to allow the state to change

Slide 14 Nested Code Blocks

- All of the different possible code blocks can be nested inside of each other. In many cases this will be the only way to build the program that you want.
- You will need to use do in order nested in a do together to have multiple sequential animation happen at the same time
- You will need multiple if statements nested inside each other to have it account for multiple possible conditions at the same time

Tips and Tricks

Slide 16 Complex Animation Tips

- If you only use do in order animations your program will feel like everything is moving on a grid
- Using do togethers you can achieve different diagonal or curved movements

Slides 17 Synchronization

- An important thing to remember when nesting different amounts of statements and wanting them to synchronize is that you can change the duration of a statemtn from the default 1 second to other durations

Slide 18 Dissolving and Converting

- You can easily convert or dissolve control blocks to maintain the current contents of the block. You can access this and more by right clicking the container.

Exercise Facilitation Step-by-Step

These step-by-step directions are for the guided facilitation option 3 that uses the Tutorial: Control Structures Do Together 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 1-4 of Tutorial Exercise

Students will open Alice and create a comparable scene or download and open the supplied world.

Talking Points

- If you are going to construct your own scene make sure you have positioned your character to face the prop and you know the distance to the object.

Module 2: Using a Do in Order

Goal – Complete Steps 5-12 of Tutorial Exercise

Students will be able to construct and predict the behavior of basic do in order functionality.

Media

- Play the video: *Using Procedures Overview*
- OR demonstrate adding procedures

Talking Points

- This is just a refresher around adding basic procedures and predicting what will happen when you execute the animation. If you need more of reminders of how to create animations you can revisit the earlier materials for adding procedures.
- The ***do in order*** control statement in Alice creates a code block for sequential execution. This control block is in every method (procedure and function) in Alice when it is first created. Look at the top corner of myFirstMethod as you build your animation.
- Be sure that your animation has basic move animations in multiple directions.

Module 3: Using the Do Together

Goal - Complete Steps 13-21 of Tutorial Exercise

Students will be able add a do together control structure and add statements to the block

Media

- Play the video: *Using Do Together.*
- OR Demonstrate adding a do together code block and adding procedures into it

Talking Points

- Alice prepares students for "thinking parallel"
- Concept: A computation may proceed ***concurrently*** (or simultaneously) at the same time with others.
- The ***do together*** control statement in Alice for "do these actions at the same time"
- This control block must be added explicitly to our code

Module 4: Using Nested Code Blocks

Goal - Complete Steps 22-26 of Tutorial Exercise

Students will be able understand why to use a do in order inside a do together to achieve the desired animation.

Media

- Demonstrate the ability to add a code block inside of code block

Talking Points

- When you have two opposite directions like moving up and moving down simultaneously execute what do you think will happen? Why?
- You can use flowcharts to help you map out what needs to happen simultaneously and what needs to happen in order for planning your program.

Module 5: Synchronizing Animations

Goal - Complete Steps 27-28 of Tutorial Exercise

Students will be able adjust animation durations to synchronize different number of nested statements.

Media

- Play the video: *Using Procedures Overview*
- OR Demonstrate adjusting the add details components of a statement

Talking Points

- As your worlds become more complicated and your animations more layered learning to adjust your animation durations to align will become a critical detail for creating smooth animations.