Alice Facilitation Guide

Alice 3 Hour of Code/Event Workshop

Summary

This project will give participants the chance to build a virtual scene, craft a simple story using a script or storyboard, translate their design into a program design, and create a 3D animation using an easy to learn programming language. This activity is a condensed version of the first 3 core Alice lessons that can be found on the alice website *Building a Scene, Programming in Alice,* and *Design Process Introduction.* You can use those lessons to familiarize yourself with the larger lessons or to augment this lesson if you have more time. The order of this lesson is altered from the *Design Process Introduction* flow to allow the participants to first get a sense of what characters and locations they will have available to them for their story to prevent them from brainstorming a story that the Alice gallery can't support. The primary goal of this lesson is to give participants a taste of 3D world building, an introduction to the design processes, an introduction to coding in Alice, and most importantly to have a good experience that motivates them to want to build more.

Lesson Overview

Times can be extended based on the amount of time of your event or workshop. This outline is based on the shorter 1 hour schedule.

- Introduction (5min)
- Scene Building Lesson and Exercise (10 min)
- Storycrafting Lesson and Storyboarding or Script Writing Exercise (10 min)
- Designing a Program Lesson and Algorithm or Flowchart Exercise (5 min)
- Programming Lesson and Animation Exercise (20 min)
    - Optional inclusion "hello, world" Exercise
- Debrief / participants Share Work (10 min)

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 participants to work together at a shared computer.

Presentation + Lecturing

Ideally, you should be able to present the lesson slides 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. This is also great for allowing participants to share their work at the need.

Supporting Materials

You may want to print out the accompanying step by step guide for use by the participants as they move through the lesson.

You will need to supply printouts of the blank storyboard templates that can be found on the accompanying webpage for this lesson, blank paper, and provide writing utensils.

You may want participants 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 and access the google docs 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 60 minutes for use in an Hour of Code or similar event but can also be extended for longer coding events or workshops by giving more time to each component.

Exercise Facilitation Step-by-Step

Activity Introduction (5 min)

Goal – Get them Excited

This module is intended to introduce the hour of code and show them what an Alice project looks like. You can play the linked Hour of code video or go to hourofcode.org and pick a relevant video.

Media

- Show a Welcome Video from the hour of code site to get them excited: https://hourofcode.com/us/promote/resources
- Present Slides 1-7

Talking Points

Slide 2 What is Alice

- Alice is a block based coding environment meaning you will not be typing code you will be using code snippet blocks. The basic components are the scene editor where you build your 3D environment and the code editor where you will program your animation.

Slide 3 An Alice Example

- You can also choose other animations from our featured projects page.
- This video has been included as a aspirational animation that can show what can be done in Alice.

Slide 4 Project Example

- For today we will focus on making a simple animation of a conversation or knock knock joke such as this example.

Slide 5 Plant the Seed

- Start to think about ideas for your animation. Inspiration for your animation could come from a joke you know, a conversation, an internet meme, or a funny comedy skit.

Slide 6 Schedule for the Day

- This is the plan for today. We will begin by building a scene, then will work on designing your animation and program, build or program, and then showcase your work for everybody (if ability to share with all).

Building a Scene (10 min)

Goal – Build a Simple Alice Scene

The goal is give a general understanding of how to use the scene editor, explore the options in the gallery for their animation, and begin to build a simple scene. You can have them follow along with Alice open or have them wait until you are done presenting.

Media

- Present Slides 1-7
- And/OR Demonstrate selecting a template and the basics of the scene editor
- Exercise Steps 1-9

Talking Points

Slide 8 A 3D Virtual Environment

- Learning to program in Alice means you are going to create 3D (three-dimensional) worlds. Three dimensions (3D) means that these worlds and objects in the worlds will have a *height*, a *width*, and a *depth*.
- The environments created for games, animations, and simulations created in 3D are called virtual environments because they allow the user to believe that they are present in that environment. This *presence* allows them to experience and interact with the different objects that make up that world.

Slide 9 Scene Templates

- When Alice starts, or File -> New is selected, Alice presents this Dialog Box that allows you to select a scene template.
- You can choose a Blank Slate or a Starter World. OR you can choose a scene you have already created under My Projects.

Slide 10 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 **Properties Panel** provides options for:
  - Changing settings for the camera window
  - Selecting objects and changing their positioning, size, color, etc.
  - Adding and manipulating unique camera and object markers
- The **Gallery** contains collections of 3D models that can be used to build a scene.

Slide 11 Move, Turn, and Roll

- All Alice objects have a *location*, which is defined by their *orientation*, and their *position* in the environment.
- All Alice objects are self-centric **–** in other words, when they move, turn, or roll in a direction, it is based on their orientation, not the viewpoint of the camera or the user.
- *Orientation* means that every object has its own sense of up, down, left, right, forward and backward.
- Typically, when Alice objects move, their position changes, but not their orientation.
- Typically, when Alice objects turn or roll, their orientation changes, but not their position.

Slide 12 The Gallery

- All the objects that can be used in Alice are found in the Gallery.
  - All objects that are going to be used in a program must be added to the Scene Editor (even if they are hidden in some way) before the program is running. It is not possible to add objects while a program is running.
- The Gallery has several different tabs that will allow you to access the models by class hierarchy, by theme, or group.
- There is a tab for searching the Gallery.
- There are tabs for shapes and other classes that participants have modified or wish to import.
- In the Biped Class you will find the Sims people builder. This can be a great way to build a custom character or personal avatar
- (Warning) Kids can spend the whole time allotted for building a scene customizing one character so be sure to pull them out)

Slide 13 Adding Objects

- In addition to dragging an object in to the scene or double clicking it to place it in the middle of the scene you can copy an object in a scene to create a duplicate by using *alt+mouse click+drag* for Windows (control+mouse click+drag for Mac OS X) on an object in the scene.

Slide 14 Naming Objects

- The dialog box will appear whenever an object is added into the scene.
- Predefined names are suggestions by Alice.
- Multiple objects of the same class will be differentiated by numbers at the end of the name (for example *queenOfHearts*, *queenOfHearts2*, *queenOfHearts3*, etc.) unless the user provides more useful names, which we would suggest.
- Rule #2, *camelCase*, is not a rule of the Alice language. *camelCase* is the convention of writing compound words or phrases with no spaces and an initial lowercase letter, with each remaining word element beginning with an uppercase letter.

Slide 15 Position with Mouse

- For more details on these options, see the How To videos on *Positioning Objects, Rotating Objects*, and *Resizing Objects*.
- (Warning) A common mistake is to select a different handle style and not put it back on the default and expect different behavior when using the mouse.

Slide 16 Position with Coordinates

- Clicking and dragging in a 3D environment can be deceiving, particularly when trying to align objects. Using coordinates allows the object to be positioned at a specific location.
- The coordinate axes are based on the orientation of the scene (ground). For example, when the camera is looking directly at the center of the ground, the coordinates are (0, 0, 0).
- Positive x is to the left as seen by the camera, and negative x is to the right as seen by the camera.
- Positive y is up as seen by the camera, and negative y is down as seen by the camera.
- Positive z is away from the camera, and negative z is toward the camera.

Slide 17 Position with One Shots

- Selecting a procedure from the one-shots menu gives the user more control in setting up the environment.
- It allows the user to set up the relative positions and distances of one object to another when creating animations (to avoid collisions, for example).
- (Warning) There is a bug that will sometimes occur that makes selecting objects in the scene stop working. You can generally save the world and restart Alice to return expected behavior but even in this state one shots will always work for manipulating objects.

Slide 18 Internal Joints

- Objects usually (not always) have internal joints or skeletons that control subparts of the model.
- A joint connects the subpart of the body to the rest of the body.
- Joints have a different orientation than the whole object, which determines how roll and turn will control them.
- The white axis points forward, red to the right, green up, blue backward.

Slide 19 Manipulate Object Subparts

- For more details on these options, see the How To videos on *Manipulating Object Joints.*

Slide 20 Camera Controls

- Clicking and holding an arrow will manipulate the location of the camera.
- Clicking and dragging in the direction of the arrow will speed up the camera movement.
- Clicking and dragging in between two arrows will combine the camera manipulations.
- For more information on Camera Controllers, see the videos on *Using Camera Controls to Change Camera Views.*

Slide 21 What is a Camera Marker

- It is a good idea to set up a camera marker for the initial camera position when starting to set up a scene. participants will often move the camera around as they are setting up the scene and may get "lost" in the 3D virtual world. This marker serves as a point of reference when this occurs.
- Markers are also useful for providing more interesting animations with multiple markers providing different viewpoints in the scene.
- For more information on Camera Markers, see the video, *Setting Up and Using Camera Markers.*

Slide 22 Save Often

- Emphasize the importance of frequently saving projects, using a versioning naming system.
- Projects can become corrupt while participants are working on them, for a wide variety of factors. Participants will not have to restart a program from the beginning if they have earlier versions of the project saved.

Slides 23 Undo / Redo

- Useful Keyboard commands:
  - Windows: Control+Z, Control+Y
  - Mac OS X: Command+Z, Command+Y

Slide 24 Creating a Scene

- First explore the gallery to see what options you have available then start to think about:
  - Where does the animation take place (choose a blank slate)
  - Choose your main characters (biped, quadrupeds, talking props)
  - What scenery items will help make for a richer scene (props)
- (Warning) Be aware that there are template starter worlds. It is ok to use these but you may want the participants to build their own scenes.

Exercise Facilitation

If you chose to use the handout have everyone complete steps 1-9

- Remind everyone to save their world first to insure backups are being made and they don't risk losing their progress
- Remind everyone to make a camera marker so that if they start to move their camera around they can return it to the desired position
- Walk the room and make sure participants aren't stuck in the Sims builder
- Remind them to explore the whole gallery
- If someone decides they want to change the location they can select the scene from the drop down and change the ground texture and sky color (don't need to start over)

Crafting a Story (10 min)

Goal – Create a Simple Script or Storyboard

The goal of this step is to learn a little bit about the design process and generate a very simple story to work.

Media

- Present Slides 26-30
- Exercise Steps 10-14

Talking Points

Slide 26 - 30 Writing a story or drawing a storyboard

- The primary objective of writing a story or drawing a storyboard is to understand:
    - Where does the animation take place?
    - What characters are involved in the story?
    - What are the key events in the story?
- Writing a script or storyboard will help you think through the key elements of the story to be animated. Either is a good way to explore and understand your story before starting to build the scene and program your animation.
    - A script is a set of directions for a director to program a play on a stage. The important parts of a script are:
        - Set design descriptions introducing the scene
        - Directions for movements around the scene
        - Specific dialogue
    - A storyboard is a frame by frame visual representation of a story used for planning animations or films. Each frame is a sketch, and sometimes includes a caption describing the intention of the frame, to communicate:
        - Visual cues about where the event is happening
        - Representations of who or what is the focus of the scene
        - Representations of or notes describing what is happening
        - The details of the sketches are minimal, with only enough to convey key story bits. The goal is to tell the entire story in the simplest visuals possible.
    - While both are great ways to better understand and flesh out the details of your story and iterate a storyboard for animation can be very helpful because it:
        - Can be quicker than writing out all of the details
        - Gives extra information about the framing of the camera that is important to think about in animations
        - Gives more visual information about the scene that can be helpful for planning your animation

Exercise Facilitation

If you chose to use the handout have everyone complete steps 10-14

- Distribute blank storyboards and pens and pencils
- Use the story starter ideas provided to help them get started
- Remind them to look at their existing scene and think about the characters and location of the scene and think about what they might be doing
- Stress that storyboards can be simple stick figure drawings
- Remind them to keep it simple at this stage. They can revisit and add more complex interactions later if time permits

Planning a Program (5 min)

Goal – Create a Simple Algorithm Outline or Flowchart

The goal of this step is to learn a little bit about program planning and think about how their story will translate into code.

Media

- Present Slides 32-36
- Exercise Steps 15 or 16

Talking Points

Slide 32 - 34 Algorithm Design

- An algorithm consists of a step-by-step list of actions that provides a description of how to perform the task.
- In Alice the objects and actions are needed for outlining the steps to be performed in carrying out the task.
- To recognize a task description, ask if this describes the plot directly in the form of:
  - Objects and actions (e.g., penguin skates)?
  - Information about how an action is performed (e.g., penguin skates slowly)
- In the story example, the objects (nouns) are highlighted in blue and the actions (verbs) in red.
  - Note that not all nouns and verbs are highlighted. We have highlighted only those that are essential to creating the animation
- The algorithm design will need to add formatting that describes how multiple actions should be performed:
  - The Do in order format indicates the instructions are to be performed in sequence
  - The Do together at the beginning of a block set of instructions tells you that they are to be performed simultaneously

Slide 35– 36 Using Flowcharts

- A common way to present programs and user experiences are different types of activity diagrams, or flowcharts. These charts visually represent the flow of the program or process (smaller part of a program) and data, as the program executes.
- Flowcharts are good for representing non-linear program flows and can also help to show more complex interactions and dependencies like simultaneous events, user inputs decision points, and repetitions.
- There are a standard set of symbols that can be used
    - Filled circle or round rectangle – show where the program (or process) begins
    - Arrows – show the directional path or flow through the program or process
    - Boxes – typically the main component of an activity diagram.
        - Show the activity that will take place at that stage in the process
        - Can represent a simple activity, or a more complex sub-process
    - Solid lines – allow you to represent concurrent activities; in other words activities that take place at the same time in the process
    - Diamond – represent decision points in the flow of the program

Exercise Facilitation

If you chose to use the handout have everyone complete either step 15 or 16

- The most important part of this step is to have them start to plan their program before diving in
- In a time pinch having them at least analyze their story to pull out the key objects and actions is a good exercise
    - Underline or circle objects (nouns)
    - Underline or list out the actions (verbs)

Coding an Animation (20 min)

Goal – Program an Animation in Alice

The goal of this module is to learn the basics of programming in Alice and to program all or parts of the designed story. This module can be as long as there is time for allowing for the final debrief and sharing of worlds. For events with more time with many participants who have never programmed before you can stop before having them begin programming their world and guide them through the "hello, world!" exercise as a way to walk them through the creation of a statement, the history of "hello, world!" and so you can pause to congratulate them all on becoming computer programmers.

Media

- Present Slides 38-36
- Exercise Steps 17-25

Talking Points

Slide 38 Defining 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 39 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: Camera View, Methods Panel, Editor, and Control Panel.

Slide 40 Methods Panel

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

Slide 41 Alice is Object Oriented

- It is important to remember that Alice is an object-oriented development environment. 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 (ex: the scene object has the atmosphere procedures and the camera has object move procedures).

Slide 42 Select the Object and Create a Statement

- The Objects Menu, located underneath the Camera View and above the Methods Panel, displays all the objects currently available to the programmer.
    - It should be noted that the “this” object listed in the menu refers to the scene object.
- To add procedures for the subjoints of an object, select the object’s SubJoint menu, 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.

Slide 43 Create a Statement

- 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.

Slide 44 Parameters

- When a procedure is added to the Code Editor, there are a series of sub-menus that represent parameters for that procedure.
- Different procedures will have different parameters:
    - *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 45 Required Parameters

- To insure that your code is always complete and functional, Alice won't let you add incomplete statements. 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 46* Optional Parameters (Add Details)**

- Almost every Alice procedure has *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

Slide 47 Control Panel

- The control panel contains tools that extend the functionality of the Code Editor.
    - The commenting tool allows you to add notes that will be ignored by the computer.
    - There are different control structures for structuring your code to execute things such as loops and do togethers.
    - 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.

Slide 48 Do In Order

- **Do in Order** is the default behavior for method tabs in the Code Editor.
- **Do in Order** treats your statements sequentially.

Slide 49 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. i.e. if you move a joint forward and backward at the same time what do you think you will see?

Slide 50 The Alice Player

- 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.
  - This is useful for testing procedures and debugging the program.

Slide 51 Incremental Development

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

Exercise Facilitation

(Optional) If you chose to have guide them through the "hello, world" exercise (http://www.alice.org/resources/exercise-and-project/hello-world/) be sure to have them save their current projects and start from a new project.

If you chose to use the handout have everyone complete either step 17 to 25

- Encourage participants to develop an incremental development style.
- Write a little code and test what is written by running it.

Advanced Topics

These are components that can be introduced into the lesson if there is enough time or can also be presented during the work sessions for those participants that are working faster than others.

Slide 53 Tips and Tricks

- This can be introduced at the beginning of the programming exercise to have them add comments from their algorithm or flowchart design into the program first before beginning to code
- Use comments to outline your program, before implementing, to make it easier to track your progress and for you and others to understand what parts of the code are intended to be implemented

Slide 54 Nesting Control Structures

- This is an extension of the do-together introduction. It should be made more clear especially if participants are going to try to make the mouth open while a say statement is being displayed.
- 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 up and down sequentially for each character while at the same time having them both happen at the same time.

Slides 55 Animating the Camera

- This is a great way quickly add a dynamic feel to an animation and is pretty easy to introduce during the work session.
- 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.
- It is important to remember to use the moveAndOrientTo not just the moveTo when using camera markers if the new position has a different orientation (pointing a different direction).

Slide 56 Scene Effects

- 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 57 Scene Effects

- Remember when setting up your scene to position objects off screen or set to 0 opacity that you want to have enter the scene or appear later in your animation

Slide 58 Adding Sound

- You can quickly make your animation come to life by adding audio to your world using the playAudio procedure
- It doesn't matter which object you attach the audio to. You will find the playAudio procedure under all objects in Alice
- There are great resources for how to do this including how to create custom audio here: https://www.alice.org/resources/alice-3-audioibrary/ All of the videos are also at the bottom of the Alice 3 how to page.
- An example word using all of these simple techniques can be found here: https://www.alice.org/featured-projects/sea-encounter/

Wrap Up - Sharing and Discussion (10 min)

Goal – Showcase the work and reflect on the experience

The end of this experience should be a celebration of the work created during the session and a time to reflect on the experience as a group. If more time is needed for programming you can easily do this while participants are still working. You can start to invite participants up to present their work whenever they are ready.

Media

- Present Slide 60
- Exercise Step 26

Exercise Facilitation

If you chose to use the handout have everyone this is reflected in step 26

- It is always fun to invite participants up to share their work on a projector. Have the participants introduce their piece with a title and any relevant information.
- For discussion here are a couple question ideas:
    - What did you like/ not like about the activity?
    - What were some of the challenges of this project?
    - Did you find it easier or harder than you thought it would be?
    - Did this change your thinking about what programming is?