Learning to program with Alice 3.x

**Contents**

Introduction

1. Introduce the Alice interface
2. Understand the different work areas

Lesson 1

1. Lean how to place an object (whiteRabbit) onto the Scene
2. Adding basic procedures (Say, and Move)
   1. Get the whiteRabbit to say messages
   2. Get the whiteRabbit to move

Lesson 2

1. Get into more detail with procedures, by making the whiteRabbit move it’s legs
2. Getting the legs and whiteRabbit to move at the same time
3. Testing the changes

Lesson 3

a) Placing the Turn procedures into a custom procedure so it can be reused

Introduction

1. What is Alice.org?
   1. Some examples of videos students created
      1. https://www.youtube.com/watch?v=cV6GYiMkmr8&index=1&list=PL8A081C21E8F99917
      2. <https://www.youtube.com/watch?v=oNma_KiLtgc&list=PL8A081C21E8F99917&index=19>
2. Explain the Alice 3 tool
   1. Launch Alice 3
   2. Select the grass template
   3. Save the project as “myFirstProject”
   4. Close and open “myFirstProject”
   5. Save as “whiteRabbitProject”

Lesson 1:

1. Open up “whiteRabbitProject”
2. On the upper left hand corner select “Setup Scene”
3. At the bottom of the “Setup Scene” are the classes
   1. Right click on the down arrow next to “all classes”
   2. Highlight “Biped classes” and then the right arrow
   3. You can see the list of biped classes
   4. Another look at the other ways to view classes
      1. Class Hierarchy
      2. By Theme
         1. Show the different themes
      3. By Group
      4. Search
      5. Shapes / texts
4. Adding the WhiteRabbit to your scene
   1. Go to Browse by “Class Hierarchy”
      1. Select “Biped Classes”
      2. Find and drag WhiteRabbit to your scene
         1. Name the WhiteRabbit (or leave it WhiteRabbit)\_
         2. Position the WhiteRabbit using your mouse
         3. Move the WhiteRabbit to the left part of your screen facing you
5. Getting the whiteRabbit to say something
   1. Click the “Edit Code” button on the bottom right hand corner of your scene
   2. Click on the whiteRabbit in your scene editor
   3. On the top of the code editor click on the **myFirstMethod** tab
   4. In the procedures tab under the scene editor drag **this.whiteRabbit say** procedure to the editor window
   5. Click on “custom textString” if you want to add anything to “hello” such as “hello there” or “hello there [your name]”
   6. Save
6. Run and test your scene and add color to the bubble
   1. In the scene on the upper right is a “Run” button
      1. Press the “Run” button
      2. Close the Scene
   2. Do you remember the color of the bubble that said “hello…”?
      1. Let’s change that color to “Cyan” or what every you like
      2. Go to the “say” function that you added and click on “add detail”
      3. Select “bubble color” and select a color
      4. Press the “Run” button and test again
   3. Do you want to have the “hello…” stay on the screen longer?
      1. Go to the “say” function that you added and click on “add detail”
      2. Select duration and set it to 2 seconds
      3. Save
      4. Press the “Run” button – do you like it?
7. Adding another text message to the whiteRabbit
   1. Click on the whiteRabbit
   2. In the procedures tab under the scene editor drag **this.whiteRabbit say** procedure to the editor window
   3. Click on the “custom TextString” and add the word “having fun?”
   4. Click ok
   5. Go to the “say” function that you added and click on “add detail”
   6. Select “bubble color” and select a color
   7. Select duration and set it to 1.5 seconds
   8. Save
   9. Press the “Run” button – do you like it?
8. Let’s add a text method using the copy and past functionality
   1. Select the text message you just added and right click and select “copy to clipboard”
   2. In the upper right hand corner of the Alice editor is a small clipboard will turn white from yellow showing something is in it.
   3. Click on the clipboard and drag it to below the second say function
   4. You now just created another function exactly the same as the last one
   5. Save
   6. Press the “Run” button – do you like it?
   7. Note by copying the function from one you already created you didn’t have to change the bubble color or the duration, it was already set from the one you copied from.
9. Change the text of the third say function
   1. Go to the last “say” function that you added and click on “having fun?”
   2. Now select “Custom TextString” and enter “Goodby”
   3. Save
   4. Press the “Run” button – do you like it?
10. Now let’s get the WhiteRabbit to disappear
    1. In the Procedures Tab for “this.whiteRabbit” – under “appearance” drag **setOpacity** procedure to after your third “say” function
    2. Select 0.0 for the Opacity
    3. Save
    4. Press the “Run” button – do you like it?
11. Let’s make the whiteRabbit turn and move between comments
    1. In the procedures tab for “this.whiteRabbit” – under “orientation” drag **turn** to the edit window under the “say – Hello …” procedure
       1. Select “turn left .25”
       2. Save
       3. Press the “Run” button
       4. The whiteRabbit should turn after he says “Hello”
    2. In the procedures tab for “this.whiteRabbit” – under “orientation” drag **move** to the edit window under the **turn** function you just added
       1. Select “forward .25”
       2. Save
       3. Press the “Run” button
    3. In the procedures tab for “this.whiteRabbit” – under “orientation” drag **turn** to the edit window under the **move** function you just added
       1. Select “turn right .25”
       2. Save
       3. Press the “Run” button
12. Can you make the whiteRabbit turn and move between the “having fun?” comment and the “good by” comment?
    1. Do you see any similarities to the steps you did in 11?
    2. Would using the copy to clipboard help you with this?
    3. After you add these functions click save, “run” and test

Lesson II – Getting the whiteRabbit to move his legs when he walks

1. Save the “whiteRabbitProject” as “whiteRabbitProject2”
2. Open up your “whiteRabbitProject”
   1. File – Save As – “whiteRabbitProject2”
3. Make the whiteRabbit walk instead of float after “hello there”
   1. Click on the whiteRabbit in the scene
   2. In the Procedures Tab, drag **turn** onto the code window and place it above the first move function
   3. Select “Forward” .125
   4. Then click “add detail” on the turn function
      1. Select duration of .25
   5. This.whiteRabbit **turn** right, .125, duration .25
   6. In the Functions Tab, drag getRightHip onto “this.whiteRabbit” of the **turn** function you just added. (you drop right on top of “this.whiteRabbit”
   7. This.whiteRabbit getRightHip **turn** Forward, .125, duration .25
   8. Press the “Run” button – does it work?
      1. Only the hip moves back, we need more functions
4. Make the left leg move
   1. Copy the last function from #2 above to the clipboard
   2. Copy the function from the clipboard to under the function from #2 above.
   3. On the function we just added, change the getRightHip to getLeftHip by selecting the dropdown arrow next to getRightHip and highlighting the right arrow next to “whiteRabbit”.
   4. Now change the **turn** from Forward to Backword
   5. Press the “Run” button – does it work? It doesn’t work smoothly…
5. Making the legs and “whiteRabbit” move at the same time
   1. Bottom left of the code window drag “do together” above the first rightHip **turn** function
   2. Drag the first two **turn** procedure into the “do together” where it says “drop statement here”
   3. Drag the **move forward** procedure into the “do together” under the two **turn** functions.
   4. Save
   5. Press the “Run” button – now does work? Does it look like a half of step?
6. Add more leg movements to the whiteRabbit to make a full step
   1. Copy the **do** **together** function to the clipboard
   2. Copy the **do** **together** function from the clipboard to under the first **do** **together** function
   3. In the second **do together** function, change the getRightHip function from **move forward** to **move backward**
   4. In the second **do together** function, change the getLeftHip function from **move backward** to **move forward**
   5. Save
   6. Press the “Run” button, looks better doesn’t it
7. Let’s put a final touch on the step
   1. Under the “whiteRabbit” functions, select straightenOutTheJoints and place it under the second **do together** function (not in).
   2. Set the duration to .25
   3. Save
   4. Press the “Run” button – this is starting to look good…

Lesson III – Creating custom methods for walking so we can reuse them

1. Instead of coping the two functions plus the straightenOutLeg function multiple times, let’s create our own “whiteRabbit” step function
   1. There are many reasons to create a function that we can reuse multiple times instead of copying the code over and over, can you tell us some reasons?
   2. On the top left of the code window there is a yellow six sided box with a dropdown arrow next to it, select the dropdown arrow
   3. Highlight “whiteRabbit” and select “add whiteRabbit procedure”
   4. In the box, name the procedure “stepForward”
   5. You now have a new procedure.
2. Add the step code to the new procedure “stepForward”
   1. Click on the “myFirstMethod” tab at the top of the code window and copy the first **do together** function into the clipboard
   2. Click on the “stepForward” tab and drag the do together function from the clipboard to the place where is says “drop statement here”
   3. Now copy the second **do together** function in your “myFirstMethod” and place it in the “stepForward” procedure under the first **do together** function you just added
   4. Now copy the straightenOutLeg function from the “myFirstMethod” and place it under the second do together function in the “stepForward” proceduere”
   5. Save the “stepForward” procedure
3. Changing the stepForward prodcedure from to this
   1. In the stepForward procedure, all of the this.whiteRabbit are red
   2. They all need to be changed from this.whiteRabbit to this
   3. This is because the procedure we created is a procedure that is only for this.whiteRabbit so it already knows this and only needs to be called this.
4. Add the “stepForward” procedure to your “myFirstMethod”
   1. In the procedures for the “whiteRabbit”, you will now see the new “stepForward” procedure you just created
      1. If you don’t see the “stepForward” procedure, make sure you have the whiteRabbit highlighted in the upper left window
   2. Drag this procedure onto your “myFirstMethod” and place it above the first **do together** function
   3. Right click on the first do together function and click on is enabled to remove the check, the function is now disabled and had lines across it
   4. Do the same thing for the second do together function disabling it
   5. Do the same thing for the straightenOutLeg function disabling it
   6. Also disable the “**move** forward .25”
   7. Save
   8. Press the “Run” button – does it work?
   9. If it works, delete the three functions in “myFirstMethod that you just disabled.
5. Let’s have the white rabbit take two steps instead of one
   1. In the procedures for the “whiteRabbit”, drag another “stepForward” procedure and place it under the other “stepForward” Procedure
   2. Save
   3. Press the “Run” button – does the “whiteRabbit” take two steps?
6. Now can you make the whiteRabbit walk from the “having fun” to the “good by”