An Introduction to Alice

This is a modification of the Shark Attack Introduction to Alice written by Teddy Ward in 2013
By David Yan, Erin Taylor, and Alex Boldt
Duke University
Under the direction of Susan Rodger
June 2015
“Begin at the beginning,” the King said, very gravely, “and go on till you come to the end”

- Lewis Carroll, Alice in Wonderland
Hello! I’m Alice, and I’m going to teach you how to use the Alice program. With Alice, you can make your own animations, using tons of different characters.
Part 1: Creating a Scene

• Our first step is to choose a background.
• When you open Alice, a box will pop up that has six different choices of background. It looks like the box to the right.
• Select the water background, because our world will be on an island.
• Click on water and then click Open.
After you click **Open**, your screen will look like this:
Saving your world

• Before we do anything else, let’s save our world. You should also always do this before you close out of Alice.

• Click on File at the top left-hand corner of your screen, and then click on Save World.
Saving your world

• In the box that pops up, name your world island, and save it in a place that you will be able to find again, such as in a folder on your Desktop.
Saving your world

• Also, while you’re working on your Alice world, this box will pop up about every 15 minutes.

• You should always click **Save right now**. This way, if Alice crashes, or if your computer crashes, you will have backups of your world and will not lose all of your work!
The Viewer

• The viewer lets you view what your world looks like:

• The arrows move the camera. **DO NOT touch them** for now as it’s hard to get the camera back to the original camera view.
Adding objects to your world

• Now, we will add some objects to the world.
• Just below the picture of your empty water world, there is a small green button that says Add Objects.
• Click on this button.
Adding objects to your world

A new screen will appear, on which there is a large selection of objects below the water screen that you can add into your world. This is called the **Local Gallery**. Each folder of objects in the gallery has a different theme.
Adding objects to your world

- Find the **Environments** folder in the gallery – you may have to scroll to the right a bit.
- Scroll to the right again until you see the **Island**.
- Click on the **Island**.
- On the box that pops up, click **Add instance to world**.
- The island will appear in the center of the ocean screen.
Adding objects to your world

The island will take up most of your viewer, which will cause problems when we want to add more objects in a minute.

To fix this, click and drag the island around the viewer until it looks like the picture on the left.
The Undo button is your friend!

• What if you make a mistake, like accidentally clicking on the ocean and moving it? Or what if your island “jumps” offscreen?
• You can click on the Undo button above the object tree to undo the last thing you did.
• Use this button whenever you mess up, or want to get rid of something you just did.
Adding a character to your world

Anyway, we’re going to add a person to our world now. Click Local Gallery above the pictures of objects to go back to the main gallery of objects.

Scroll to the right until you see a folder called people. Click on it.
Adding a character to your world

- Click your favorite person from the gallery and then click **add instance to world**.
- Don’t spend too long choosing a person!
Positioning the objects

- **Click and drag** your character into the middle of the island as shown. We’ll get him/her unstuck from the sand in a second.
Positioning the objects

• Look at the right side of your screen.
• There is a group of buttons with faces on them that are used to position objects.
• We want to have our person stand on the island, so click **Move objects up and down** (the 2\textsuperscript{nd} one)

![Image of buttons](image)

• Click on your character, hold, and move your mouse up until his/her feet just touch the sand.
• Bonus: move the island down into the water a little bit (move the character down again, too)
The Object Tree

• When you add objects to your world, they will appear in a list on the left of your screen, called the Object Tree.
• You should see both your island and your character there.
• If you want, right click on your character’s name, select rename, and type any name you want. I renamed mine “heroine.”
• Some objects have parts.
• For example, if you click on the plus mark next to the island in your object tree, you’ll see that the palm tree and its fronds and coconuts are all objects in your world.
• These are fixed in position, unless you move them with *methods*. 
Object Parts

- Right click on coconut1, choose methods, move, down, other..., and then a calculator will pop up for input.
Object Parts

• Type 2.25 into the calculator, and when you press okay one of the coconuts will fall to the ground.
• Turning, rotating, and moving object parts is a good way to make more complex animations look more realistic!
• Note: “move” may detach a part from its object but “turn” and “roll” may move parts without detaching them
Adding objects to your world

• Now return to the Local Gallery.
• Scroll over to the Vehicles folder and click on it.
• Scroll over to Rowboat and add that to your world.

Your world should look like this:
Positioning the objects

- Now move the rowboat to the front left of the island using the first three buttons in that group on the top right.
- Click **Move objects freely** (the 1\textsuperscript{st} one) to move it around.
- Click **Move objects up and down** (the 2\textsuperscript{nd} one) to lower it into the water.
- Click **Turn objects left and right** (the 3\textsuperscript{rd} one) to align it with the shore.
Nice job setting up the scene! Now we’re going to learn about the camera!

(If you have extra time, add more scenery! Use the Ocean folder in the local gallery to add shells, fish, etc. Don’t add a shark just yet, though 😊 )
Part 2: The Camera

• A *Dummy camera* is like a camera tripod – it saves the location of your camera view. This way, if you move your camera around, you can always get back to a certain position by moving to a dummy camera location.

• Look to the right side of your screen, and find a gray button under your object positioning buttons labeled **more controls**
The Dummy Camera

- More buttons will appear after you click more controls.
- Click on the button that says drop dummy at camera. It will seem like nothing happens, but don’t worry, and only click the button one time.
The Dummy Camera

• Once you have clicked this button, a folder will appear on your object tree labeled **Dummy Objects**.
• If you click on the **plus sign** next to the **Dummy Objects** folder, a list of your dummy camera positions will appear.
• Right now, there is only one position, called **dummy**.
The Dummy Camera

• Whenever you add a dummy camera position, you should rename it so that you know which position it is.
• Right-click on dummy in the object tree, and then choose rename. Type in STARTview.
• Similarly, change the folder name Dummy Objects to CameraViews
• You should add a dummy at your starting camera position whenever you start a new Alice world.
Moving the Camera

• Now that we have a dummy camera set up, we can move the camera freely without losing our place.
• There are three sets of arrows beneath the scene that move the camera.
Moving the Camera

The first set moves the camera up, down, left, and right.

The second set moves it forward and backward and turns it from left to right.

The last set rotates the camera up and down.
Moving the Camera

- Click and hold the **left arrow in the second group** to turn the camera left until the island is out of the picture
- Dragging your mouse farther from the arrows moves the camera faster
- Be sure not to move the camera too far up or down
The plot thickens...

- Click **Add Objects**, and navigate to the “Ocean” gallery
- Click and **drag a shark** up to the window to add it to this part of the world
The plot thickens...

- Move the shark using the positional buttons so that it’s half in the water and roughly facing the island
- Click more controls, then drop dummy at camera so that we have this view saved
- Rename this dummy camera “SHARKview” in the object tree
Moving the Camera

• To restore the camera, right click on camera in the object tree.
• On the menu that pops up, choose methods, then camera set point of view to, then CameraViews, then STARTview.
Finishing Setup

• Now just click **Done** (in green) towards the bottom right!
Now that we are done setting up our camera, we can start to animate the characters in the world!
Part 3: Methods

• The large beige rectangle in the center of your screen is called the **Method Editor**. Right now, it is blank.
Methods

• The method editor is where you can make your characters do things.
• Your characters already know how to do certain things.
• These are some of the things that your character already knows how to do. To find this list, click on your character’s name in the object tree. Then look below the object tree at the box that says details, and click on the methods tab. This list will appear.
Methods

- To program your character to do something, click on one of these methods, hold down your mouse, and drag and drop it into your method editor. Let’s try dragging in **say** to start. Select **other...** to be able to choose what we want he/she to say. Then type something like “Hey! Welcome to my island!” and press **okay**.
Methods

• Now press the **Play** button in the upper left-hand corner of the screen to watch your AMAZING program in action!!!!!

• Okay. That was pretty boring. Let’s spice things up by teaching our character to do a backflip
• Notice that there is no method for backflip created yet!
Methods

• To teach your character new things, you can combine methods that he/she already knows into new methods.

• Make sure you have clicked on your character in the object tree. Then, go to the methods for your character and click create new method.
Backflip Method

• In the box that pops up, type **backflip**, then click **OK**.

• You should see a new tab appear in your method editor called **heroine.backflip** (heroine will be replaced by your character’s name). This is the space where you will program the backflip.
Backflip Method

- Drag **heroine move** into the method editor
- Select **up**, then **1 meter**
- Drag another **move** below this, but this time select **down** then **1 meter**.
Backflip Method

- To finish the flip, drag in **turn** between the two movements (a green line should appear when you’re between them)
- Choose backward, then 1 revolution (all the way around)
- Now your character will move up, flip, and then move down
Backflip Method

• Your code should look like this:
Events

- Now that we have written a method, we’ll use an event to see it in action.
- The event editor is found in the top right-hand corner of your screen.
- Events are used to tell our program when it should call our methods.
Events

• You’ll see that when the world starts (when you press Play), your world does world.my first method.
• Click where it says world.my first method, and change it to your character > backflip.

- Now when you press the Play button in the upper left, your character will do a backflip!
Backflip Method

• Depending on how tall your character is, he/she might have hit his/her head on the ground. To fix this, click **1 meter** next in the **move up** command, and choose **other**... Try typing **2** in the calculator that comes up to start.

• Be sure to adjust the **move down** command, too.
Backflip Method

• Finally, to make have our character say “Hey! Welcome to my island!” and then do the backflip, select the `world.my first method` tab again
• Drag in `backflip` from your character’s list of methods.
Testing!

• Now change your “When the world starts” event in the event editor back to world.my first method.

• We do all this because testing one method at a time will be a really good time-saving habit later.

• Now just press to run the whole thing!
Part 4: Add a Surprise

- First, drag in a **shark say** to the bottom of world.my first method. Select other, then type “S” in the box that comes up.
- Click **more...** next to this in your method editor, a menu of properties will appear. Select **duration**, and change it to **2 seconds** so people have more time to read the message.
- Then, drag in a **heroine say** to the bottom of world.my first method. Select other, then type “What is S?” Change the duration to **2 seconds**.
- Your method will look like this:
Changing Camera Views

• S also stands for “Shark,” so we’ll have the camera get a look at the shark.
• To get the camera to change position while your program runs, we have to use its methods.
• Select camera in the object tree, then drag in its set point of view to method.
• Select CameraViews > SHARKview on the menu that comes up.
Adding a Surprise

- Your world.my first method should look like this:

```plaintext
world.my first method No parameters

No variables

heroine say Hey! Welcome to my island! duration = 2 seconds more...

heroine.backflip

shark say S duration = 2 seconds more...

heroine say What is S? duration = 2 seconds more...

camera set point of view to SHARKview more...
```
Adding a Surprise

• Drag in a **shark say** to the bottom of world.my first method. Select other, then type **“S is for SURPRISE”** in the box that comes up
• Click **more**... next to this in your method editor, a menu of properties will appear. Select **duration**, and change it to **2 seconds**
Your method should look like this:

```plaintext
heroine ▼ say Hey! Welcome to my island! ▼ duration = 2 seconds ▼ more... ▼

heroine.backflip

shark ▼ say S ▼ duration = 2 seconds ▼ more... ▼

heroine ▼ say What is S? ▼ duration = 2 seconds ▼ more... ▼

camera ▼ set point of view to SHARKview ▼ more... ▼

shark ▼ say S is for SURPRISE ▼ duration = 2 seconds ▼ more... ▼
```
Moving Camera Views

• Now we want to move the camera back to the starting position.

• Select **camera** in the object tree, then drag in its **set point of view to** method.

• Select **CameraViews > STARTView** on the menu that comes up.

• The line of code is shown:
Adding a Surprise

• Then select **shark** in the object tree, drag in a **move towards** method, and choose **2 meters > rowboat > the entire rowboat**
• This will have the shark move 2 meters towards your boat.
Adding a Surprise

• Change the distance of your move towards method to how far away you think you estimate your shark is from the rowboat.
• To do so, click amount=2 meters, other, then type in your guess on the calculator.
• Set the duration (under more...) of your move towards method to 5 seconds so the user has time to react to the shark.
Adding a Surprise

• Your method should look like this:
Test

• Now press **Play** to test your world!
• Your shark should stop just in front of the boat, but because we just guessed his distance to the boat, he probably won’t. Play with the **distance** in your **move toward** call until your shark ends up like this:

![Shark and boat image]

• For me, the distance was 4 meters.
Moving the Boat

- We’d like to move the rowboat with the arrow keys
- Click **create new event** in the event editor (top right)
- Choose **Let the arrow keys move subject**
- Change **camera** to **rowboat > the entire rowboat**
Moving the Boat

• Try running the animation now and moving the boat with the arrow keys
• You will see that it moves by itself
• We’d like to move the boat with the Heroine inside, so we’re going to create a GetInBoat method to accomplish that
Part 5: GetInBoat Method (Using Vehicles)

- While the shark can’t exactly walk on land and attack our character, we’re going to have him/her escape in the rowboat anyway.
- Click on your character in the object tree, then create a new method. Call it `getInBoat`
Testing the getInBoat method

• To test the getInBoat method, change the event “When the world starts” to call the heroine.getInBoat method
• This way we can test only the getInBoat method as we develop it.
Now click on the **properties** tab in the details pane (where the methods are)

The properties pane lets you change various things about your object.

Later, you’ll be able to make your character red, radioactive, or invisible

For now, look at the **vehicle** property
GetInBoat Method

- Drag the **vehicle** property into the method editor (make sure the new **getInBoat** tab is selected) and select **rowboat > the entire rowboat**.
- Now when the rowboat moves, your character will move with it; your character is riding the boat.
GetInBoat Method

• Your method should look like this:
Moving the Boat

• Try running the animation again and moving the boat
• You will see that the heroine is now moving *with* the boat *but not in it.*
• You should notice that the boat can go off the screen.
• We would also like the camera to move with the boat as well.
Setting the Camera’s Vehicle

• To fix this, repeat the steps from before to set the camera’s vehicle to the rowboat.
• Now your camera will follow the boat as it moves around.
• Your method should look like this:
Moving the Boat

- **Play** your world! You should notice another problem:

- Your world may look like this, so now we need to place the heroine inside the boat.
Editing GetInBoat

• Drag in a **move to**, and select **rowboat > the entire rowboat** (to move your character to the boat)
• Drag in an **orient to** (scroll down), and select **rowboat > the entire rowboat** (to face the front of the boat)
Do Together

• Now, from the bottom of the screen, where there are several advanced (but common) coding constructs, drag in a *do together*.

• Normally, actions in Alice take place *in order*. One line of code will run, then the next one.

• By putting multiple methods inside of a *do together*, those methods can happen at the same time.
Do Together

- Now we can put all of our methods into the **Do Together** block.
- Order does not matter, since they will all run at the same time.
Editing GetInBoat

• Drag in a **say above** the do together, and have your character say something like “Oh no! Use the arrow keys to help me escape!” so that the person using your world knows what to do.
GetInBoat

• Finally, drag in a **move** *below* the do together, select **down > ½ meter.**
• This will make your character sit more realistically in the boat. The exact distance (or even direction!) may be different for different characters – experiment!
Back to world.my first method

• Go back to the event space, and change the “when the world first starts” event back to calling the world.my first method

• We will now be editing world.my first method

Events

When the world starts, do

Let move

heroine.getInBoat

my first method

rowboat heroine
Calling GetInBoat

• Select your character in the object tree, and drag your new method, `getInBoat` into your `world.my first method`. It should look like this:
Do together

• Now, from the bottom of the screen drag in a do together.
Do Together

- Drag our last two lines into the do together
- Otherwise, our character would wait until AFTER the shark attacked to get in the boat
Updated Method

Your world.my first method should look like this:

```
world.my first method No parameters

No variables

heroine say Hey! Welcome to my island! duration = 2 seconds more...
heroine.backflip
shark say S duration = 2 seconds more...
heroine say What is S duration = 2 seconds more...
camera set point of view to SHARKview more...
shark say S is for SURPRISE duration = 2 seconds more...
camera set point of view to STARTview more...

Do together
shark move amount = 4 meters toward target = rowboat duration = 5 seconds more...
heroine.getInBoat
```
Play your world! Everything should be working correctly
callForHelp method

• Now we want to let our heroine call for help
• Click on your **character** in the object tree, then create a new method. Call it “callForHelp”
• In the method just drag in a heroine say, select other, and type “HELP”
• Your method should look like this:

```
heroine.callForHelp No parameters
```

```
No variables
```

```
heroine \(\triangleright\) say HELP! \(\triangleright\) more... \(\triangleright\)
```
callForHelp method

• In addition to screaming “Help!” we also want our heroine to wave her arm at the same time.
• To have our heroine do these things at the same time, drag in a “Do Together” block.
callForHelp method

• Now we want to access our heroine’s shoulder in order to make her raise it.

• In the object tree, click on the plus sign beside the heroine and select the left shoulder.

• Note: Our arm movement uses the “leftShoulder” object for our character but this can change depending on which character you choose. For many, it might be called “leftArm.”
callForHelp Method

• Now we want to add code so that her arm moves up and then back down.

• To do this select the left shoulder, then select the turn method and turn the heroine’s left shoulder $\frac{1}{2}$ revolution.
callForHelp Method

• Now that we have moved our heroine’s arm up, we must move it back down.

• Follow the same steps from the previous slide, except turn her shoulder forward this time. The code should be as follows:

```javascript
heroine.callForHelp No parameters

No variables
```

- Do together

  - heroine say HELP! more...
  - heroine.leftShoulder turn backward 0.5 revolutions more...
  - heroine.leftShoulder turn forward 0.5 revolutions more...
callForHelp method

• Run the callForHelp method by changing the “When the world starts” event to have it run heroine.callForHelp

• Run your world. Does it work?
callForHelp method

• Notice that your character only says “HELP!” and doesn’t raise his/her hands
• Why not?
• Notice that the two arm movements are in opposite directions and are both in a “do together” block
• Therefore they cancel each other out!!
• How do we fix this?
Fixing callForHelp method

• We need the arm movements done in order, BUT at the same time as when she yells “HELP!”
• To do this, we need the arm movements in a “do in order” block nested inside the “do together block” with the “HELP!” say method.
Fixing callForHelpMethod

• Next, we drag in the arm movement methods into the “do in order” block.
• Remember, we want the arms to go backwards first, and then forwards, so put them in that order.
Test callForHelp method

• Run your world now
• The heroine should now put her arm up while yelling “HELP!”, then put her arm back down
Finishing callForHelp method

• We need to change the “when the world first starts” event back to call the world.my First method

• Do as follows:
Adding an Event

• We want our heroine to be able to call for help on command.
• To do this we are going to create a new event.
• Select **create new event** in the event editor (top right), then choose **When a key is typed**.
Adding an Event

• Change **any key to letters > H** and **Nothing to callForHelp**.

• Note that this will call the “callForHelp” method **ANYTIME** you press H, but your users will only know to do so when you instruct them.
Calling the Event

• To let the user know that they can press H to make the heroine call for help add in a [Heroine say] in the world.

• Here is the call:

Now when you run the world, you should be able to make the heroine call for help whenever you press H
Circling

• Right now, the shark just sits by the beach like a chump after you ride your boat away

• We’ll have it swim circles around the island instead to keep things exciting
Circling

- Open up world.my first method via the object tree.
- Click on shark in the object tree, and drag shark think into the method editor. Select other, and type in something like “Rats. I’ll just have to wait here.”
Circling

• Drag in **shark turn**, then select **left > other > 10 revolutions**.
• Under **more...** choose **asSeenBy > island > the entire island**.
• This will make the shark turn around the island instead of just spinning in circles
• Change the **duration** to **50 seconds**
Circling

- Play your world.
- If the shark circles very close to the island, move your boat offshore a bit in the viewer (why?)
- If the shark doesn’t circle the island, copy the code below (note your mistakes, though!)

```plaintext
heroine \_\_\_ say Press H to make me call for help! \_\_\_ duration = 3 seconds \_\_\_

shark \_\_\_ think Rats. I'll just have to wait here. \_\_\_ more...

shark \_\_\_ turn left \_\_\_ 10 revolutions \_\_\_ asSeenBy = island \_\_\_ duration = 50 seconds \_\_\_
```
Comments

• Now note the **double slash** on the far right side of the line of coding constructs at the bottom of your editor.
• This icon lets you add *comments* to your code that won’t affect the program when it actually runs.
• Drag and drop one anywhere in your code and write an explanation for the line below it. An example:
If you later want to unglue your character from the rowboat, set the \texttt{vehicle} property of your character from rowboat back to \texttt{world} (be sure to do the same thing for \texttt{camera}!)
Finished world.my first method

<table>
<thead>
<tr>
<th>world.my first method</th>
<th>No parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>No variables</td>
<td></td>
</tr>
</tbody>
</table>

- **heroine** say **Hey! Welcome to my island!**
- **heroine.backflip**
  - **shark** say **S** **duration = 2 seconds**
  - **heroine** say **What is S?** **duration = 2 seconds**
  - **camera** set point of view to **SHARKview**
  - **shark** say **S is for SURPRISE** **duration = 2 seconds**
  - **camera** set point of view to **STARTview**

- **Do together**
  - **shark** move **amount = 6 meters** **toward** **target = rowboat** **duration = 5 seconds**
  - **heroine.getInBoat**

| heroine **say** **Press H to make me call for help!** **duration = 3 seconds** |
| shark **think** **Rats. I'll just have to wait here.** |
| shark **turn** left **10 revolutions** **asSeenBy = island** **duration = 50 seconds** |
Congratulations! You have just made your first Alice world. There are many more things that you can do with Alice, so keep exploring it!
Extensions (in order of awesomeness)

• Make both of the heroine’s arms move up and down when she calls for help
• Add more scenery.
• Make the coconuts fall from the tree when you press spacebar.
• Make your boat your favorite color (hint: you’ll have to change the color AND texture).
• Figure out how to change the weather; make a foggy sunset.
• Check out the “functions” tab (next to “methods” and “properties”) and do something interesting with them.
• Dream up many more worlds, where magical, impossible things happen. If you ever need help, try out some more try out some more tutorials on Duke’s Alice webpage.

“It’s important to have specific dreams. Dream Big. Dream without fear.”
- Randy Pausch, creator of Alice