Treasure Hunt
A Point & Click Adventure Game in Alice

By Elizabeth Liang
under the direction of
Professor Susan Rodger
Duke University
July 2010

Step 1: Setting Up the World

The first step is to set up the world. This means positioning all the objects that we will need in our story.

There are two scenes in this game, an island and a dock. We will set up the island scene first then the dock scene. If you want to have more than two scenes in your game, review the scene change tutorial to learn how to add more scenes.

Create a new world with a water template.

This game does not have the superGround.a2c object imported unlike the scene change tutorial because we will only be using the water ground. If your game requires different ground textures, you will want to see the scene change tutorial so see how to do this.

Step 1: Island Scene Objects

Add an island, sailboat, and jock to the world.

Position the objects roughly as shown in the image.

Drop a dummy at the camera. Rename the Dummy Objects folder in the object tree to cameraViews. Expand the folder, rename dummy to islandView.

If you cannot find an object in the gallery, use the Search Gallery tool.
Step 1: Lantern & Light bulb

Add a light bulb and a lantern. You’ll notice that the light bulb disappears into the center of the island.

Right click on the light bulb in the object tree, select methods, move to lantern, flame. This will move the light bulb to the lantern so that we can see it.

Light bulb moves to:
- lightBulb move toward
- lightBulb move away from
- lightBulb orient to
- lightBulb turn to face
- lightBulb point at
- lightBulb set point of view to
- lightBulb set pose
- lightBulb stand up
- lightBulb set color to
- lightBulb set opacity to

As seen by:
- camera
- light
- ground
- cameraView
- island
- sailboat
- jock
- roof

Candle holder:
- bar1
- bar2
- bar3
- bar4
- bar5
- bar6
- bar7

Step 1: Lantern & Light bulb Continued

Resize the light bulb and move it so that it is in the center of the lantern.

Glue the light bulb to the lantern by clicking on the light bulb in the object tree and going to its properties. Set the vehicle property of the light bulb to the entire lantern.

Move the lantern so that it is above the jock’s head and looks like it is being hung in the coconut tree on the island as shown.

Step 1: 3D Text

Create two 3D Text objects, one that says Congratulations! and another that is just an ‘X’.

Rename them in the object tree and position as shown.

Create 3D Text

Add a ToyBox2 object to the island, resizing it and positioning it on top of the X.

If an object is not visible when you add it, click on the object in the object tree and move it up 1 meter to help find it in the scene.

Step 1: Island Arrow

In the Web Gallery, find the Triangle object in the Shapes folder. If you do not have access to the web, you can use a triangular prism colored red.

Right click on the triangle in the object tree and roll it left ¾ of a revolution. Position it in the upper left hand corner of the screen.

Rename the triangle in the object tree to islandArrow.

Congratulations!
Click on the X 3D Text object in the object tree and find the properties panel. Set the color of the X to black. We are finished setting up the island scene.

Set the isShowing property of the sailboat, jock, toyBox2, and the two 3D Text objects to false. This is because we only want those objects to show up later.

Add a Beach Terrain object into the world. Switch to quad view to position it.

You want to turn/move in the from the top view so that the beach looks like shown. The beach should cover the horizon line between the water and the sky in the dock single view. The island scene view should still have open water behind it. Change the camera view back and forth to check until it looks correct.

Turn the camera left ½ revolution from the object tree to find a space to set up the new scene.

Drop a dummy at the camera to save the view. Expand the cameraViews folder and rename dummy to dockView.

To switch between views, simply right click on the camera and set the point of view to the other view.

Add a Pier and Lighthouse to the scene. Use the from the top view to move them into the dock scene and then single view to position them.

Put the lighthouse on the right hand side of the beach and turn the pier so the wide end is facing the camera. Move the pier to the left hand side of the screen.

Also add a rock to the scene. The rock will initially be small. Move the rock to the pier and then resize it and move it to the right, in front of the lighthouse in the water.
**Step 1: A Pirate Map**

The next thing to add is a treasure map. Since there is no treasure map object, we will use a billboard.

Either download the pirate-map.gif from the website for this tutorial or find your own image of a treasure map.

Make a Billboard of this image and set it on the pier as shown in the image to the right. **Hint:** Try moving the billboard to the pier and then moving it up and forward.

Now we will hide this map behind a sailboat.

---

**Step 1: Dock Arrow**

Click on jock2 in the object tree and set the vehicle of jock2 to sailboat2.

Add another Triangle and position it in the upper right hand corner of the dock scene.

Rename it dockArrow.

**Step 1: Another Boat**

Add another instance of the sailboat and jock into your world. Move the new sailboat (sailboat2) in front of the map. **Resize** the boat so that the sail covers the map entirely, hiding it from view.

Move the new jock (jock2) to the boat in the dock scene.

We use two copies of the same object in the different scenes to make it appear that the boat moves from the dock to the island when in reality one set is ‘disappearing’ and the other set is now able to be seen. Sometimes it is easier to trick the viewer with copies of objects rather than figure out a complicated animation to move the objects.

---

**Step 1: Light Beam**

To create a beam of light coming from the lighthouse, we will use the flashlight object in Alice. Add a flashlight into the world.

When moving the flashlight to the lighthouse you will need to resize and check in quad view that the flashlight is inside the uppermost part of the lighthouse. The body of the flashlight should be covered as much as possible by the lighthouse with only the beam shining outward.

Set the isShowing of the flashlight to false. You are now done setting up the world!
Step 2: Change Scene

The first thing to do is to create a `changeScene` method that we can call when the arrows are clicked on. Create a new world level method called `changeScene`.

In this method, create a new object parameter named `cameraView`. Create a new color variable called `skyColor`.

Step 2: Change Scene Continued

The first thing to do in the `changeScene` method is to save the current color before we fade out.

Drag the `skyColor` variable into the method. Set value to black. Select set value to black. Set duration to 0 seconds.

Click on `world` in the object tree and select the `atmosphereColor` property and drag it over the black box in the code.

Step 2: Using Fog

The next step is to fade out of the scene. Drag the world `atmosphereColor` property into the code and set it to black. Change the duration to 0 seconds.

Drag the world's `fogStyle` property into the code and set it to `density`. Change the duration to 0 seconds. This activates the fog in the Alice world which we will use to cover our scene so that it looks like it is fading out.

Finally, drag in the `fogDensity` property into the code and set it to 1. Set the duration for 2 seconds.

Step 2: Change Scene Continued

Click on `camera` in the object tree and find the camera set point of view to method. Drag this into the code. Select expressions, `cameraView`.

Change the duration to 0 seconds.

By using the `cameraView` object parameter, we will be able to easily switch between scenes without needing to rewrite code.
Step 2: Using Fog Continued

To finish up the method and have it fade in to the new scene, reverse the commands we dragged in earlier. Drag in a `set fogDensity to 0` command with a `duration` of 2 seconds.

Set the `atmosphereColor` back to the `skyColor` we saved earlier and set the `fogStyle` to `no fog`; both commands have a `duration` of 0 seconds.

Above is the final code for `world.sceneChange`.

Step 2: Change Scene Events

Create two new ‘When the mouse is clicked on something’ events.

Set the events to happen when you click on each arrow.

Events

- `world.my first method`
  - When the world starts
    - `world.changeScene cameraView = dockView` -

- `world.my second method`
  - When `is clicked on` islandArrow
    - `world.changeScene cameraView = dockView` -

Change the `do Nothing` into `world.changeScene` and select the appropriate `cameraView`. Make sure you click on the objects in the `cameraViews` folder and not the island or the dock itself!

Test this by playing your world and clicking on the arrows.

Step 3: List Visualization

Move the `listVisualization` into the dock scene and bring it forward until the wooden border just touches the bottom of the screen.

Move it off screen so that you can only see half of it as shown.

Set the `vehicle` of the `listVisualization` to the camera and set the `isShowing` to false.

List visualizations are one of the tools in Alice to help understand how lists work. In this case we will be using it to line up the objects that the user picks up in the game.

By setting the vehicle to the camera, we ensure that the objects follow the player around. This is a useful trick for games where you want the user to have ‘items’.

Step 3: List Visualization Continued

The next thing we will do is allow the player to pick up certain objects. To do this we will use a ListVisualization.

In the Visualizations folder, add a ListVisualization object into your world. To initialize the list, click new item once. Then click OK to add it to the world.
Create a new **world** level method named `pickUp`.

In this method, create a new **object** parameter named `object`. This will represent the object that we want to pick up.

Drag the **object** parameter into the code after the `if` statement and select `object` say.

Have the object say ‘just a regular’ and set the duration to 2 seconds.

Create another object say command after the `else` and have it say ‘you have picked up a’. Again, set the duration to 2 seconds.

Make sure that there is an extra space after the last word in each string.
Go to the world functions tab and under string, drag the a joined with b function into the code over the two phrases the object says. Select default string for b.

Now drag the what as a string function over ‘default string’ and select expressions, object. The method is now finished!

Create two new When the mouse is clicked on something events.

When the lightBulb and pirate-map are clicked on, the world.pickUp method should be called on those objects.

Create a new world variable called storeAtmosphere.

The type should be color.

Drag the new variable into world.my first method and set the value to black.

Drag the world atmosphereColor property over the black box and set the duration to 0 seconds.

Play your world. Change to the island scene and click on the light bulb in the lantern. The light bulb should move to the bottom of the screen and say ‘you have picked up a lightBulb’. Change back to the dock scene, the light bulb should travel with you and when clicked on say ‘just a regular lightBulb’. Now we will work on the story starting with my first method.

If the lightBulb moves off screen, check that the vehicle of listVisualization is set to camera. Also try turning the listVisualization ½ revolution to the left in the set up.
Step 4: My First Method Continued

Drag the world `atmosphereColor` property into the code and select black. Set the duration to 0 seconds.

Add in `island say` commands to give the user instructions on how to interact with the world. Remember to set the `duration` so the text displays for long enough.

This is the final code for `world.my first method`. Your directions may vary.

Step 4: Testing My First Method

Before we test we need to change the camera view back to the island. Right click on the camera and have it set point of view to the `islandView`.

Watch the animation and change the duration of the instructions if needed.

Step 5: Lighthouse On

Create a new world method called `lighthouseOn`.

This is the method we will call when we click on the lighthouse. We want the lighthouse to turn on if the user has picked up the `lightBulb`, otherwise we want to give a hint about what to do.

Drag in an `If/Else` statement and once again drag the `listVisualization contains item` function into the conditional. Select the `lightBulb`.

Step 5: Lighthouse On Continued

We want the `lightBulb` to disappear so drag the `lightBulb isShowing` property into the `If` portion and set it to `false`.

We also want the flashlight to appear. Drag the `flashlight isShowing` property into the `If` portion and set it to `true`.

Now to lighten the sky, drag the world `atmosphereColor` property into the code beneath the other commands and select expressions, `storeAtmosphere`.

If `lightBulb isShowing` is `false`, set `lightBulb` properties to:
- `color` = 
- `opacity` = 1 (100%) 
- `vehicle` = lantern 
- `fillingStyle` = solid 
- `position` = position

Else (Do Nothing)
If the lightBulb is not in the list, we want the user to receive a hint. Drag in a jock2 say command into the Else portion of the code. Have him say ‘We’ll need that lighthouse to work to sail out’. Set the duration to 2 seconds. The lighthouseOn method is now complete!

Create a new ‘When the mouse is clicked on something’ event. When the lighthouse is clicked, lighthouseOn should happen.

To make it more interesting, we will have the light beam turn around and round in the lighthouse.

Drag the flashlight into the Events Editor and select When the world starts. This will create an event under the heading ‘flashlight’.

An object event, like an object method, should only involve the object. If you save the object that the event is tied to, the object’s events will go with it. Since this event only involves the flashlight, it makes sense to have it as an object event.

Right click on the new When the world starts event to change it to a While the world is running BDE event.

Drag a flashlight turn command into the During part of the BDE. Have it turn left 1 revolution.

Set the duration to 2 seconds and the style to abruptly so that there is no pause between each rotation.

Play the world again and get the lighthouse to light up. You should see the beam of light should be spinning the whole time the world is running.
Step 6: Sailboat Move

Once the lighthouse is on, we want the sailboat to be able to move to the island when it is clicked on. If the lighthouse is not on, then we want the jock to give another hint.

Create a new world level method called sailboatMove.

Drag an If/Else Statement into the new method.

Drag the flashlight isShowing property into the condition of the If statement.

Drag a sailboat2 move forward 10 meters command into the If portion of the code, this will move the sailboat off screen.

Drag in a Do together and set the sailboat2 and jock2 disappear by setting the isShowing property for both to false. Make the sailboat and jock on the island scene appear by setting their isShowing property to true. Change the duration for all four commands to 0 seconds.

Step 6: Sailboat Move Event

In the Else portion, have the jock2 say ‘It’s too dark to navigate’. Set the duration to 2 seconds.

We’re now done with this method!

Create a new ‘When the mouse is clicked on something’ event and have the sailboatMove method happen when sailboat2 is clicked on.

Step 6: Testing Sailboat Move

Play the world. Try clicking on the sailboat without the lighthouse on. The jock should give the hint.

Turn the lighthouse on and move the boat away. It should allow you to click and pick up the map. If the map is too big you may need to resize it.

Go back to the island scene and make sure you can now see the jock and sailboat in that scene.
Once the jock is on the island, we want to be able to 'give him' the map so the X will appear and the treasure can be found. If the user hasn’t picked up the map, we want the jock to give a hint.

Create a new world level method called findX.

Drag an If/Else Statement into the new method.

Under the listVisualization’s functions, drag the contains item function into the conditional. Select the pirate-map.

Set the isShowing of the pirate-map to false and the x to true in the If portion.

In the Else section, have the jock say ‘I wish I had a map’ as a hint. Set the duration to 2 seconds.

Create a new ‘When the mouse is clicked on something’ event and have the findX method happen when the jock is clicked on.

Test the world by clicking on the jock on the island with and without the pirate map.

Once the X is revealed, the user will click on it to show the treasure and end the game. We have one more method to write to do this.

Create a new world level method called showTreasure.

Drag in the toyBox2 isShowing property and set it to true. Then drag in a jock say command and have the jock say ‘There’s the treasure! Thanks for your help!’ Set the duration to 2 seconds. Finally drag in the congratulations! isShowing property and set it to true. We’re almost done!
Create a new 'When the mouse is clicked on something' event and have the showTreasure method happen when the x is clicked on.

These are all the events in this world. You will see that nearly all of them are click events, because this is a click-adventure game.

Congratulations on finishing this tutorial and creating a click-adventure game!