Fluency with Alice

Workbook to Accompany Snyder's *Fluency with Information Technology, 4th Edition*

by Robert Seidman, Philip Funk, Jim Isaak, Lundy Lewis

Chapter 1. The Alice Environment and Alice Worlds

In Chapters 1 and 2 we introduce you to the Alice environment and to Alice worlds. We do so by creating a short story as well as by developing an animated movie to tell the story. Together, we will set the scene, place characters in the scene, and then ask them to perform a script. Our end product will be an animated movie of Act 1 of our play.

Our cast of characters includes a young woman named Robin and a penguin named Peter. Act 1 of our play opens with Robin and Peter standing in front of a farmhouse. Robin notices Peter and moves toward him. This startles Peter and Peter's reaction in turn surprises Robin.

Tech Talk – A play or movie director would refer to the characters, Robin and Peter, as roles. In Alice, they are called "objects" & their names appear bolded & in lower-case (**robin** & **peter**).

In Chapter 1, we set the stage for Act 1. To see the animation that you will create by the end of this chapter, watch this **Video Demo** (sound on): Video 1-1. By the end of Chapter 2, you will have completed the entire Act 1 animation. {**Video Demo**: Video 1-2} [Internet connection and Flash player required. See Appendix A for details.]

In this Workbook, your role is movie director. This means that you not only direct the actors, but you also direct the camera operator and the props people.

1.1: Setting the Stage

"All the world's a stage, and all the men and women merely players." – As You Like It, Act II, Scene 7 – William Shakespeare

Let's set the stage by selecting the background scenery and by selecting and placing our characters (i.e., Alice objects) on the stage. We also mark the stage so that we can make sure that our characters are in their proper places when we roll the camera. The scene we create will look like the picture shown in **Figure 1-1**. The director, working from a script, first sets up the scene that is the basis for our play. In **Figure 1-1**, you see that the scene is shot in an open field of grass with a farmhouse in the background. In front of the farmhouse are two characters, a penguin named Peter and a young woman named Robin.

Figure 1-1 Director first sets up this scene.

Steps to set the stage

STEP 1 - Launch Alice

From the Alice folder, launch Alice. [See Appendix A.] You will see the opening screen shown in **Figure 1-2**.

Figure 1-2 Opening screen when Alice starts up.

STEP 2 - Create the World

In the Welcome to Alice! window shown in **Figure 1-3**, make sure that the Templates tab is open by clicking on it. Each template provides a ground surface and a sky color for the Alice world you are about to create. The script calls for a farmhouse in a grassy field so click on the grass template to select it. Then click on the Open button to insert the grass template as the background scene for your world. [You can also get the Welcome to Alice! Window by File/New World.]

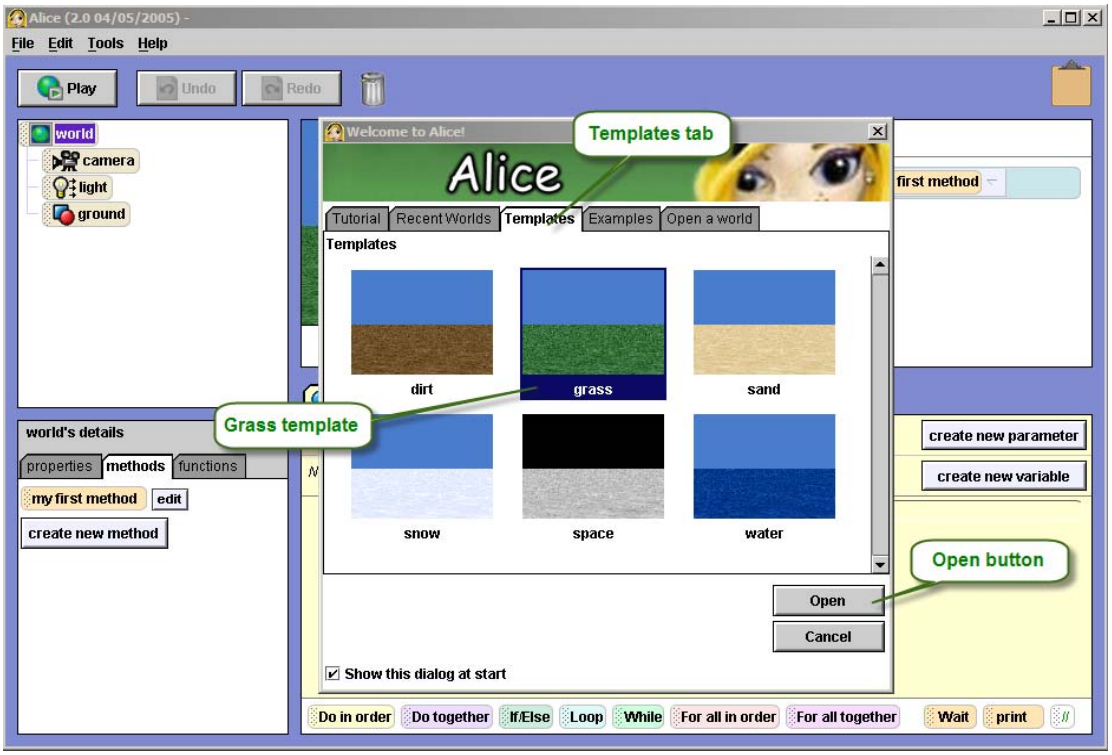


Figure 1-3 Welcome to Alice! window.

Next you see a screen like the one in **Figure 1-4**. This screen displays the Alice development environment with which you will set up and manage your Alice World. A world is a container for your characters and props - and your directions for the roles these objects perform. Notice that this screen is divided into a number of views that we call panels. Each of these panels is briefly explained below.

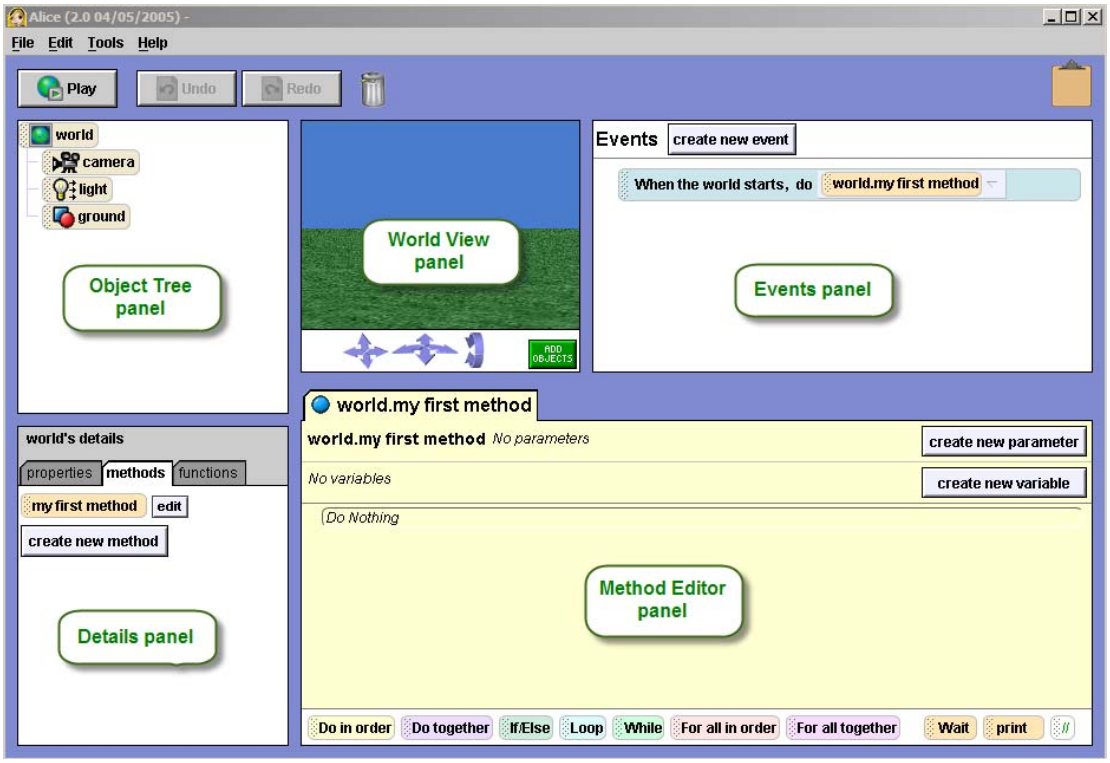


Figure 1-4 Alice development environment.

World View panel: Shows the template scene you have selected for your world and any things and characters (called objects) you place in it.

Object Tree panel: Shows the objects (things & characters) in your world. All worlds contain a **camera** object, a **light** object, and a **ground** object by default.

Details panel: Shows the details for any one object that is selected in either the **Object Tree** or the **World View** panels.

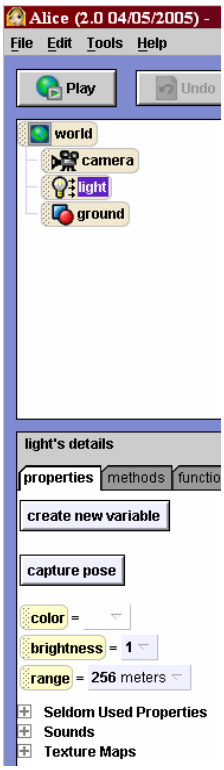
Events panel: An event is something that happens to trigger behavior in a world. Clicking the  button in the upper-left-hand corner starts (i.e., plays) the world - it triggers the When the world starts ... default event.

Explore: You can easily resize the panels by using your mouse to click on and drag the blue borders between panels.

Explore: Try examining the details for an object in your world. Click on the light object in the Object Tree panel to select it and then take a look at light's properties and methods in the Details panel by clicking on the tabs. The light's properties are things that the light object knows about itself. And the light's methods are things that the light object knows how to do. See **Figure 1-5**.

Method Editor panel: This is where you provide directions (i.e., instructions) to objects in your world which they follow when you click the Play button. Until you provide directions, nothing happens.

So far there are three objects in the world: **camera**, **light**, and **ground**. Click on the **light** object. Click on the properties tab in the **light's Details** panel. See **Figure 1-5**. Notice that the **light** object has three properties: *color, brightness,* and *range*. Click on the down-arrow next to the 1 in *brightness* and change it to 0.25. Notice that the **world** in the **World View** panel dims accordingly. Click the Undo button at the top of the screen in the Toolbar area and the *brightness* returns to level 1.

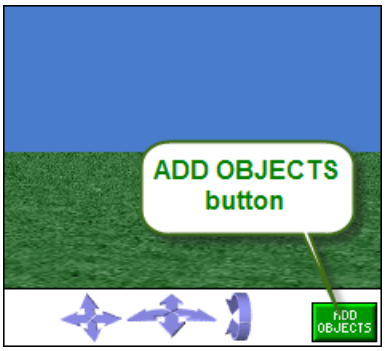


Alice Tip: The Undo button is your friend. If you make a mistake and want to rectify it, just click on Undo. It's good when you are experimenting with Alice.

Figure 1-5 light object's properties.

STEP 3 - Populating the scene

It is time to set the scene for our animated movie. Let's place a farmhouse on the open field of grass.



Click on the ADD OBJECTS button located in the **World View** panel. See **Figure 1-6**. This will open the object Gallery at the bottom of the screen in place of the **Method Editor** panel. The Local Gallery (i.e., as part of Alice on your computer) is shown in **Figure 1-7**.

The Local Gallery is where you can find things (i.e., objects) to add to your world. The kinds of things you can add are organized into a large number of different categories. You can find even more categories online at http://alice.org.

Figure 1-6 ADD OBJECTS button.

Figure 1-7 Local Gallery categories.

STEP 4 - Add a Farmhouse to your world

Each category tile in the Local Gallery contains a collection of objects that we can include in our world. Click on the Buildings category. The Buildings category opens. Scroll to the right until you find Class FarmHouse. Click on it and then click on the Add instance to world button as shown in the picture below. {**Video demo**: Video 1-3}

Tech Talk: Think of classes as factories – they make things called objects. For example, the Penguin class knows how to make penguin objects. When you add an instance of the Penguin class to your world, you add a penguin object.

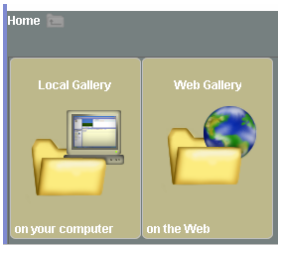
STEP 5 - Add a penguin to your world

Click the yellow up arrow in the Search Gallery panel as shown in the following picture to move up a level and return to the Local Gallery. Click on the Animals category to open it. Scroll to the right and click on Class Penguin. Click on the Add instance to world button.

STEP 6 - Add a girl to your world

Click the yellow up arrow again to return to the Local Gallery. Scroll to the right to the People category and click on it. Click on the Class AliceLiddell (Alice Liddell was the real-life name of the little girl that Charles Dodgson, aka Lewis Carroll, modeled his Alice in Wonderland books after). Click on the Add instance to world button.

When finished, your screen should look like **Figure 1-8**.



Note: If you are connected to the Internet, there is a Web Gallery you can use directly from Alice. Click on the yellow up-arrow enough times to see the Web Gallery tile. Click on the tile to have access to the Web Gallery. We are not asking you to use objects in the Web Gallery at this time – but we will in Chapter 3.

When you are finished with the Gallery, click on the DONE button on the right-side of the screen, as shown in **Figure 1-8**, to exit the ADD OBJECTS gallery. The **Methods Editor** panel will be restored.

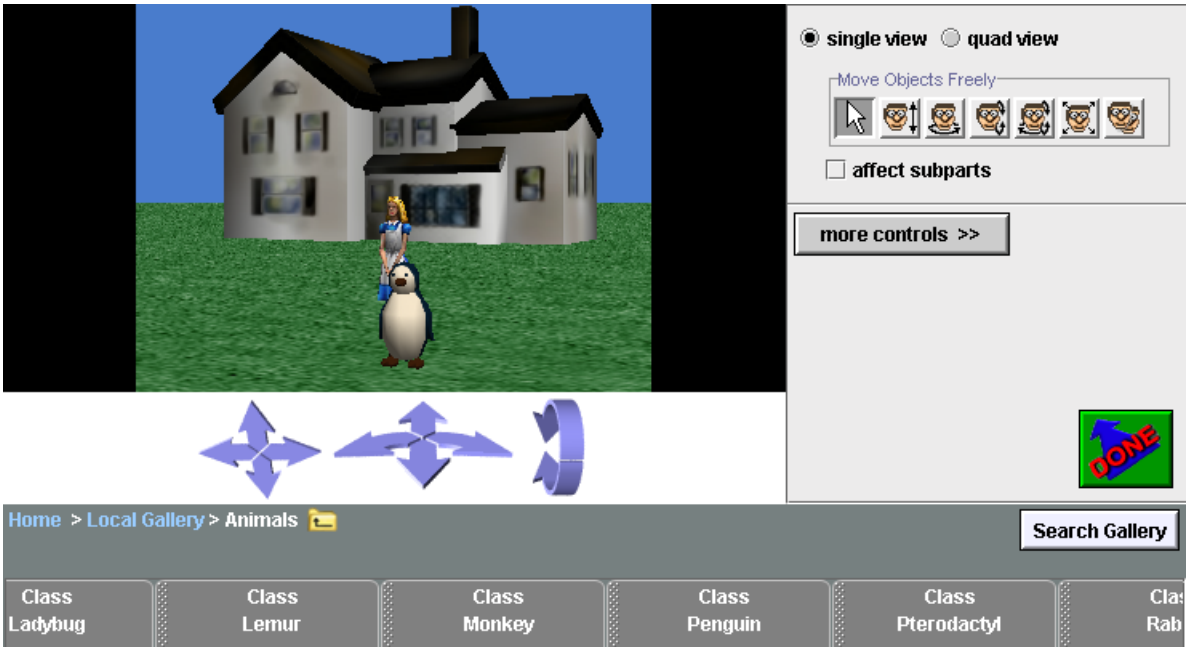


Figure 1-8 Grass world with farmHouse, penguin, and aliceLiddell objects in default positions.

STEP 7 - Actors, take your places

As the movie Director, you must move the actors to their places in the scene. The **farmHouse** is fine where it is. But, the **penguin** object and the **aliceLiddell** object need to move to their respective starting positions.

But first, let's give our actors names we want them to have for this play or movie. We prefer to refer to them by the names we give them rather than the default names assigned by Alice. We now begin to use the standard object naming convention which starts object names with lower case letters. We also **bold** object names in this Workbook just to make them stand out. We bold panel names too.

Suggestion: If you were preparing a document for a course, you know that it is a good idea to save your work periodically. Perhaps that's a good thing to do now. In the Menu bar at the top of the Alice window, click on File and then on Save World. This is no different from saving any other file on your computer. Alice worlds are saved with an *a2w* extension. Alice will regularly prompt you to save your work. Good to always take the advice!

Tip: We suggest saving your Alice world files in a folder for a course you are taking or in some other easily identifiable folder on your computer so it is easy to find. Good file management habits will pay off many fold. File/New World will bring up the Welcome to Alice! window that you see in Figure 1-3. You can locate your most recent files in the Recent Worlds tab.

Tech Talk: You can open an Alice world file only from the Alice software itself (File/Open World). Unlike Word files and most other file types, you can't just double-click on the Alice file icon stored on your hard-drive to open it.

In the **Object Tree** panel, right-click on **penguin** and choose rename in the pop-up window. With the mouse cursor, select **penguin** and then type in "peter." Press the Enter key. Next, right-click on **aliceLiddell** and choose rename. With your mouse cursor, select **aliceLiddell** and then type "robin." Press the Return key. You have just renamed the two objects. {**Video demo**: Video 1-4}

We are now going to move **robin** and **peter** to their starting positions on the grass. We will put down marks on the grass and then tell them to move to those marks. We use the marks so that if and when we have to reshoot the scene we can direct the actors to return to their original starting positions.

First, in the **World View** panel, separate **robin** and **peter** as shown in **Figure 1-9** by clicking on one object at a time and dragging it to the side. Notice that a box will appear around an object in the **World View** panel when you click on it. {**Video demo**: Video 1-5}

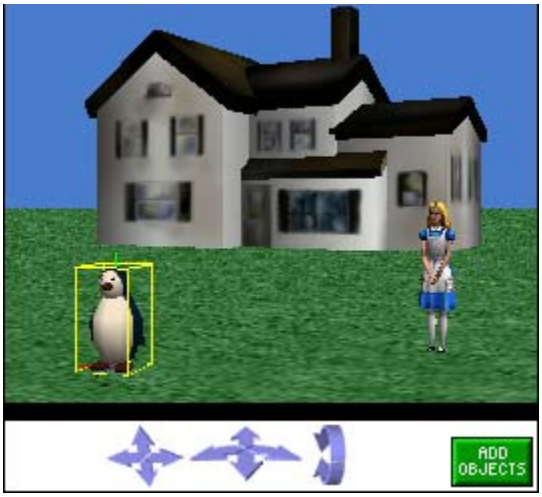


Figure 1-9 Separate peter and robin. peter is selected in this Figure.

We are now going to place cone markers where we want **peter** and **robin** to stand. Later, we will make the cones invisible to the audience. Think of this as placing large X tape marks on the stage where we want the actors to stand.

Click on the ADD OBJECTS button again. You will see the Gallery bread crumb trail. If necessary, click on Local Gallery in 

. Then, in the Local Gallery at the bottom of the screen, scroll to the Shapes category and click on it. Click on the Class Cone and on Add instance to world. With your cursor, move this cone to **peter's** right. Note that from your perspective (i.e., looking at **peter**) you move the cone to your left. But, from **peter's** perspective – the cone is placed to his right. See **Figure 1-10**. Click on Class Cone again and Add instance to World. With your cursor, move it to **robin's** left, - which is to your right. See **Figure 1-10**. Click on the DONE button to exit the ADD OBJECTS screen.

Figure 1-10 Marker cones set up.

Previously, we renamed **penguin** to **peter** and **aliceLiddell** to **robin**. In the same way, rename the cone nearest **peter** to **peterCone**. Then, rename the cone nearest **robin** to **robinCone** Note that we continue to follow the recommended naming convention - start object names with lower case letters, no spaces between words, and capitalize the first letter of subsequent words in the name.

These cones are now placed where we want our actors to stand when the movie opens. It's now time to move **peter** and **robin** to their starting positions as identified by their marker cones.

In the **Object Tree** panel in the upper left of the screen, right-click on **peter** and select methods/**peter** *move to*/**peterCone**. See **Figure 1-11**. Watch **peter** move to his cone in the **World View** panel.

{**Video demo**: Video 1-6}

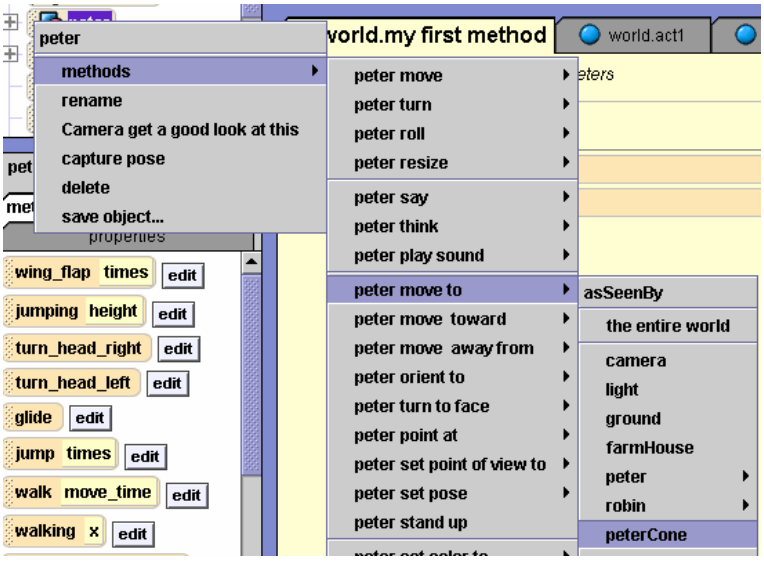


Figure 1-11 Move peter to his position at peterCone location.

Do the same for **robin**. In the **Object Tree** panel in the upper left of your screen, right-click **robin** and select methods/**robin** *move to*/**robinCone**. Watch **robin** move to her cone.

> *Tech Talk:* The things that peter and robin can do are defined by their methods which come preset from the factory, so to speak. Later, we will create custom-made methods for our characters. Here, we asked them each to exercise their built-in move to methods by sending them a message, much like you might ask someone to do something by sending him or her an IM or mobile phone text message.

Because the cones are just positioning marks, we want to make them invisible to the camera and thus invisible to us. In the **Object Tree** panel, click on **peterCone** to show **peterCone's** details in the **Details** panel. Click on the properties tab in **peterCone's Details** panel. Then, click the number you see to the right of the *opacity* property and choose invisible. To make the **robinCone** object invisible, click on **robinCone** in the **Object Tree** panel and then in the **Details** panel properties tab, click on the number to the right of *opacity* and select invisible. {**Video demo**: Video 1-7} By the way, you can easily change the cones' color by clicking on the color property.

Let's again demonstrate how these place markers can be used. But this time, the markers (i.e. cones) are invisible to the camera. In the **World View** panel, use your mouse cursor to move **peter** and **robin** closer together. See **Figure 1-12**.

Figure 1-12 Cones are invisible to the camera. peter and robin are closer together.

Now we will have our characters take their places at the invisible cone markers. In the **Object Tree** panel, right-click **peter** and select methods/**peter** *move to*/**peterCone**. Then, right-click **robin** in the **Object Tree** panel and select methods/**robin** *move to* /**robinCone**. Both characters should now be at their starting positions and the director should be happy. See **Figure 1-13.** Save your work.

Figure 1-13 Characters at their starting points as viewed through the camera.

> *Object Insight:* Notice that we did not move either peter or robin ourselves. Instead, we sent them a message asking them to move themselves to a particular location just as the Director would do.

STEP 8 - Camera Point of View

The director has set the stage, placed the actors, and now needs to properly place the **camera** in order to begin the action for this scene.

Point of view (aka, POV) is an essential concept in writing and theater. For movies, and for Alice, the story point of view is controlled by you, the Director, via the **camera** object that comes standard with every Alice world.

In Alice, you have the ability to reposition the **camera,** as well as any other object. The **camera** is really your eyes looking at the scene's physical setup. It is from the **camera's** point of view that your audience, and you as the Director, sees the action that takes place. Take another look at **Figure 1-13**. This is precisely what the **camera** sees; nothing more, nothing less.

Suppose that you are looking through the **camera** lens at **Figure 1-13**. Let's have the **camera** point at just **robin**. In the **Object Tree** panel, right-click on the **camera** object's icon and select methods/**camera** *point at*/**robin**/the entire robin. See **Figure 1-14** and the result in **Figure 1-15**.

{**Video demo**: Video 1-8}

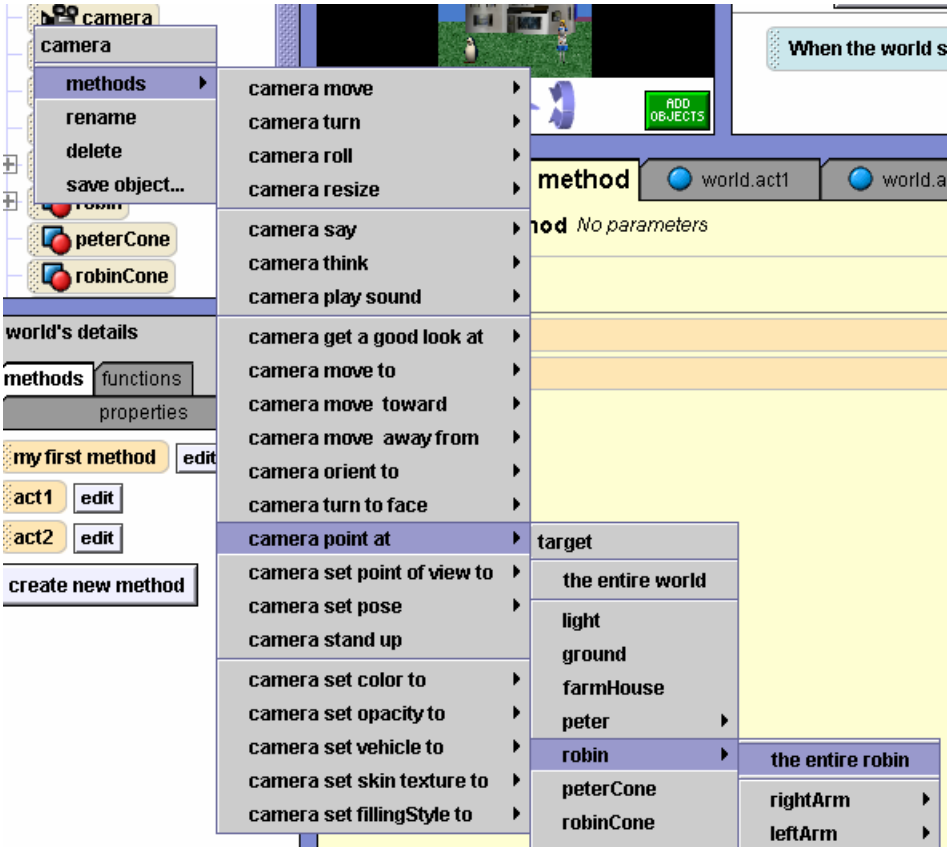


Figure 1-14 Commands to point camera object at robin object.

Figure 1-15 camera pointing at robin (entire object).

Click the Undo button to return to the previous **camera** position as seen in **Figure 1-13**.

Our first scene calls for two camera positions: Position #1 is the wide-angle camera view that you see in **Figure 1-13**; position #2 will be a close-up of **robin's** face. We can create a **camera** marker by using a dummy object which does for the **camera** what the cones did for **peter's** and **robin's** initial positions. In other words, we want to set positions for the **camera** just like we set positions for our two actors.

If the Object Gallery is open in the bottom of your screen, leave it open. If it is not open, click on the ADD OBJECTS button. In the upper right corner of the screen, click on the more controls >> button. As a result you will see the additional controls as shown in **Figure 1-16**.

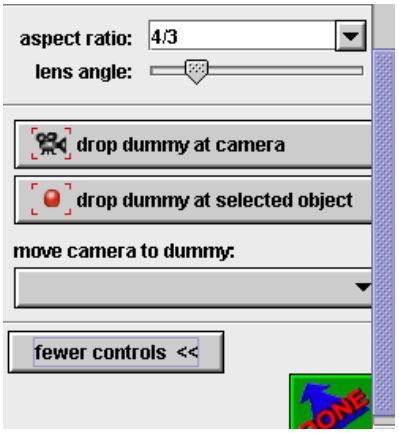


Figure 1-16 Additional controls.

To create a marker at the current **camera** position (i.e., a wide angle shot of the entire scene layout), click the drop dummy at camera button. {**Video demo**: Video 1-9}

Notice that a Dummy Objects folder appears in the **Object Tree** panel. See **Figure 1-17a**. Click on the + sign to the left of the Dummy Objects folder and notice that the **Dummy** object is present. See **Figure 1-17b.**

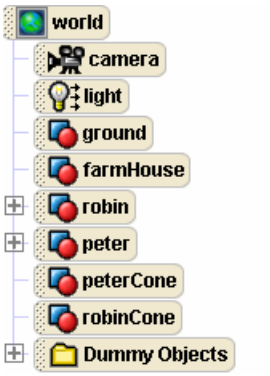


Figure 1-17a Dummy Objects folder in object tree.

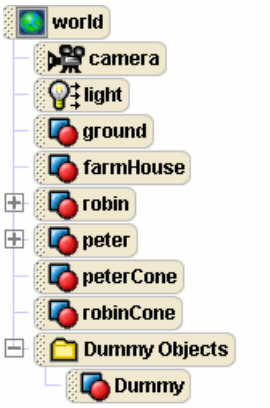


Figure 1-17b Expand Dummy Object folder to see Dummy object.

Rename the **Dummy** object to **wideAngleShot** by right-clicking on it and selecting rename in the pop-up window. Notice that we are following the standard naming convention of not including spaces in object names. Save your world.

Let's move the **camera** in for a close-up of **robin's** face so we can drop another dummy object at that camera position. Right-click on **camera** in the **Object Tree** panel. Select methods/**camera** *turn to face*/**robin**/**neck**/**head**/the entire head. The **camera** is now turned to face **robin's head**, but this is not quite the close-up we want.

By clicking and/or holding your left mouse button down on the middle of the three camera controls  at the bottom of the **World View** panel, you can move the **camera** until you have a close-up of **robin's** face. See **Figure 1-18**. Note that hovering your cursor over a camera control will reveal what that control does. Use the Undo button if you want to start again. {**Video demo**: Video 1-10}

Figure 1-18 Close-up of robin's face. Cursor hovers over the middle camera control.

When you have an acceptable close-up of **robin's** face, click the drop dummy at camera button. In the **Object Tree** panel, rename the new **Dummy** object (found in the Dummy Objects folder) to **robinCloseUp**. Click on DONE to close the Object Gallery

Move the **camera** back to the wide-angle position by right-clicking on the **camera** in the **Object Tree** panel and selecting methods/**camera** *set point of view to*/Dummy Objects/**wideAngleShot**. See **Figure 1-19.**

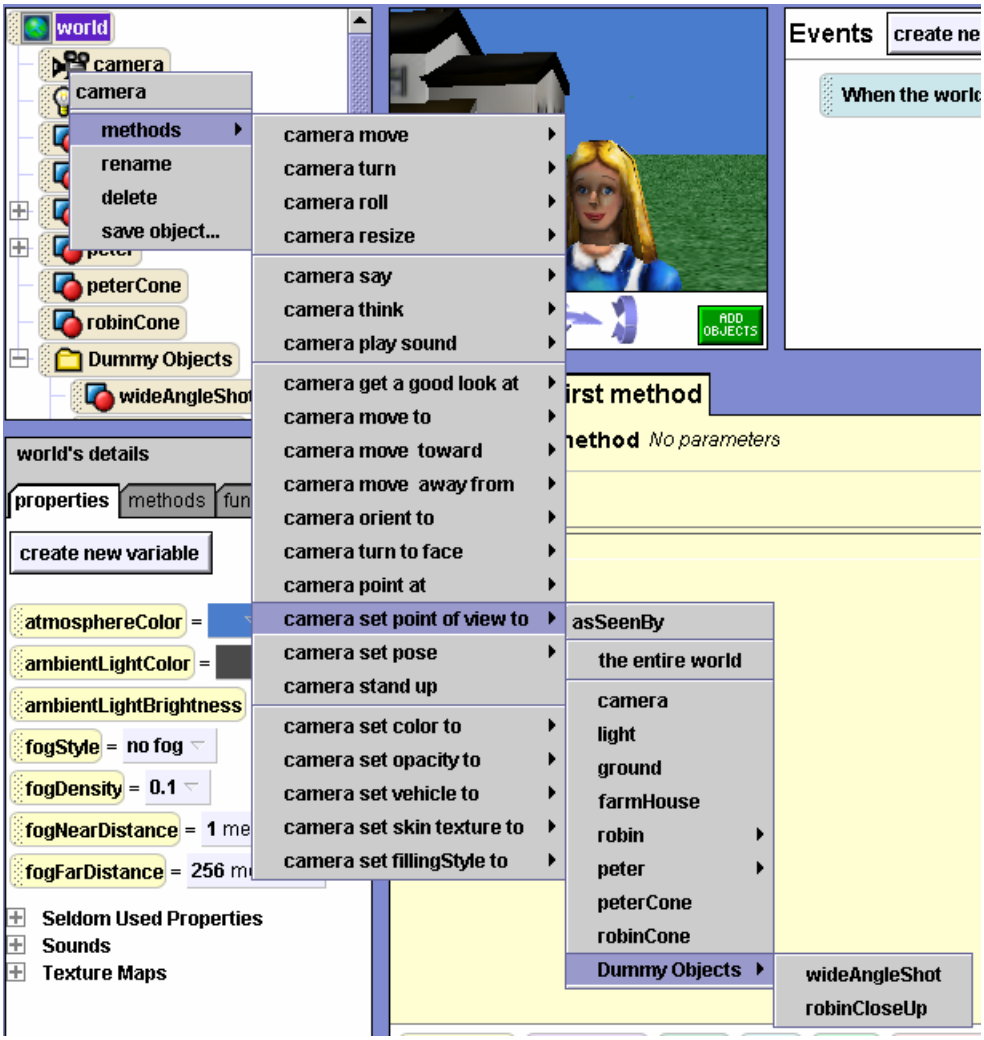


Figure 1-19 Move camera POV back to the wide-angle shot.

You now have two camera markers: a wide-angle point of view and a close-up point of view of **robin's** face. This might be a good time to save your work again

Now that our characters are in their places, we are ready for action! Click the DONE button if you have not already done so. Notice that the Gallery is replaced by the **Method Editor** panel which shows the default method *my first method*. See **Figure 1-20**.

> *Tech talk*: The full method name you see is world.my first method. "my first method" is a method name designated by the creators of Alice. The reason you see world.my first method is because Alice adds "world" connected by a period to indicate the name of the object that owns and can implement the method. This holds true for all object methods: object.method-name.

1.2: Lights, Camera, Action

"Things must be as they may" – Henry V, Act II, Scene 1 – William Shakespeare

It's Action time!

Now that we have our actors (i.e., objects) in their places, it's time to give them their directions for Act 1. This is done in the form of messages sent to the actors to tell them to perform certain actions (i.e., their methods). Notice that in the **Events** panel, in the upper right of the screen, there is a statement that says When the world starts. When you click the Play button, the **world** executes the method named *my first method*. The event When the world starts is similar to the director calling out "action." From the computer's perspective, it is equivalent to commanding: "computer – begin processing the instructions you find in the method."

The details of the **world**.*my first method* are shown in the **Method Editor** panel in the lower part of your screen. See **Figure 1-20**. Currently it says *Do Nothing*. It is in this panel that we will create the "acts" into which we will place the directions to our actors. For now, just click on the Play button in the upper left corner of your screen. The world will open in the World Running… animation window but nothing happens because there are no directions in *my first method* at this time. Close the animation window.

> *Tech talk*: The Play button is the way we tell Alice to execute the program that we have created. A program is a set of instructions (aka, directions). Something has to start the action and this is the way that Alice starts the action.

Creating the Acts

Look at **Figure 1-20**. The scene is set with the actors (i.e., **peter** and **robin**) and the props (i.e., the **farmHouse**) in their starting places. The **Events** panel is set to carry out whatever directions we place in the **world**.*my first method* method by virtue of the When the world starts event that is shown below.

But, since our story is in two Acts and up to now we only know the story in Act 1, we need a container in which to place the Act 1 directions to keep them separate from the directions for Act 2. This is easy to do, as shown in the following steps.

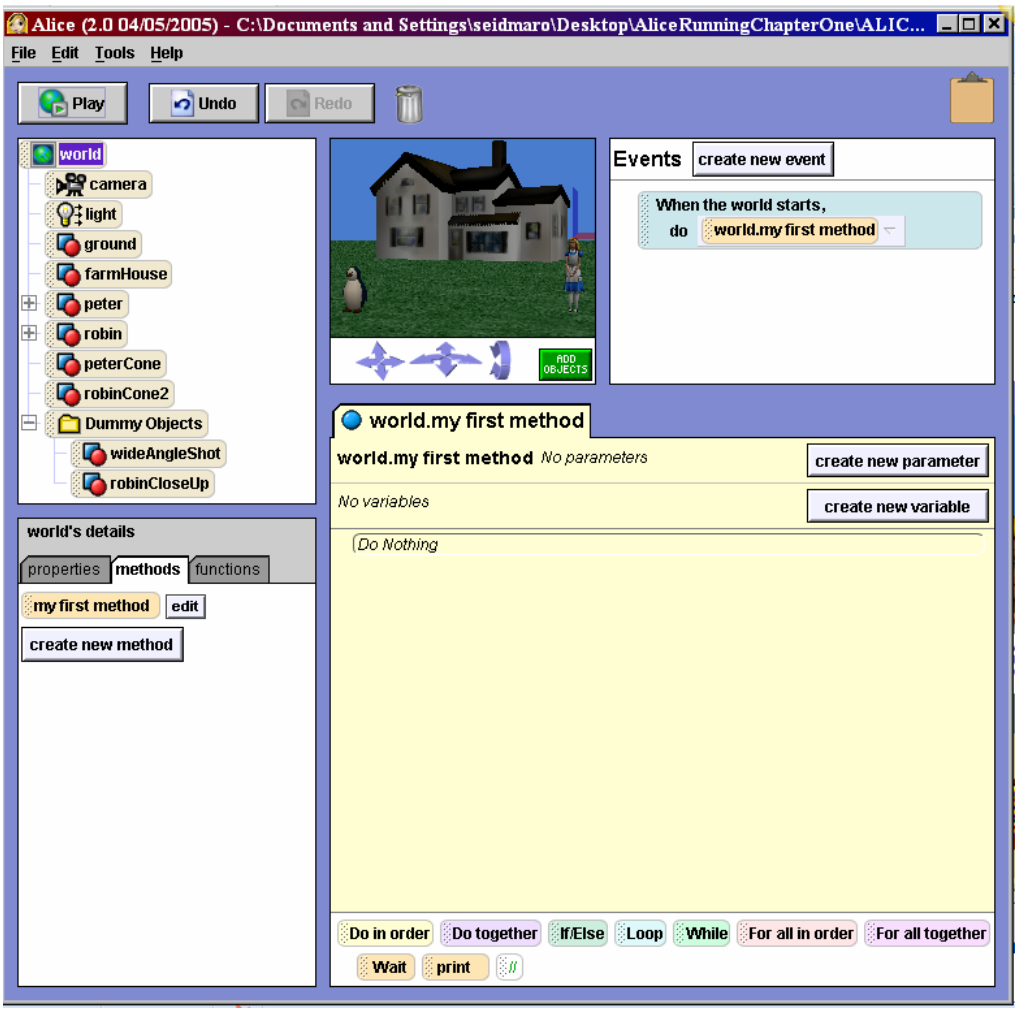


Figure 1-20 Wide-angle camera position. Method Editor panel showing my first method.

STEP 1 – Select world object

Click on the **world** object in the **Object Tree** panel to be sure that it is selected.

STEP 2 – Create new method (act1)

In the **world's Details** panel click on the methods tab. Then, click on the create new method button. Type "act1" into the New Method window text box. Click the OK button. See **Figure 1-21**. {**Video demo**: Video 1-11}

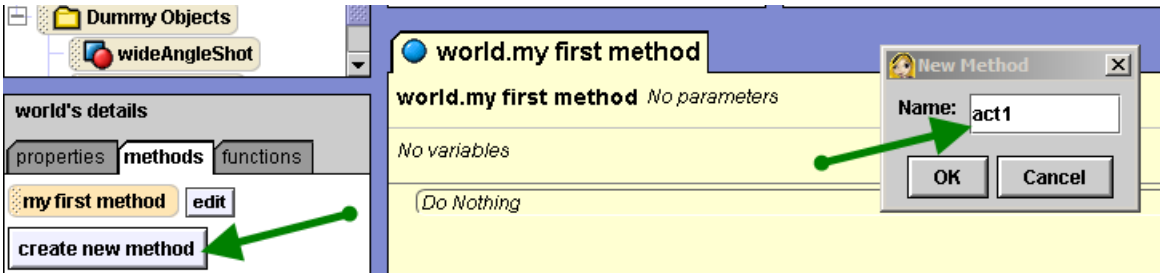


Figure 1-21 Create a new method called act1.

STEP 3 – Create new method (act2)

Repeat Step 2 to create *act2*. The results of doing Steps 2 and 3 are shown in **Figure 1-22**.

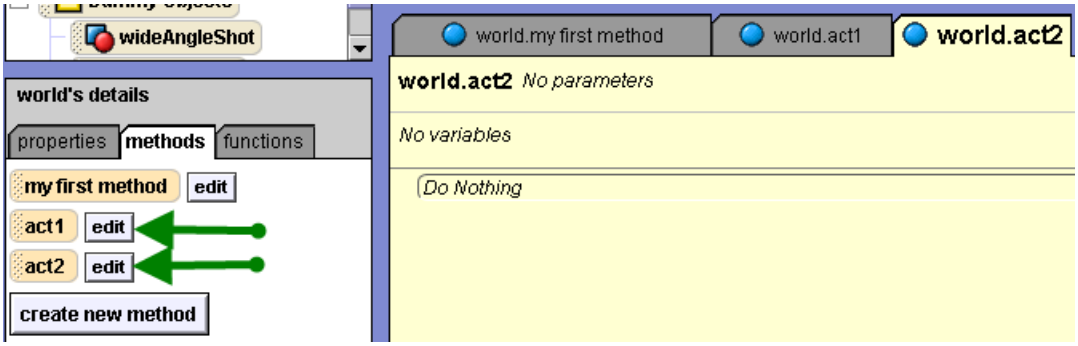


Figure 1-22 The two act methods.

STEP 4 –Edit world.my first method

Click on the **world**.*my first method* tab in the **Method Editor** panel to select it. See the tab at the top of **Figure 1-23**.

STEP 5 – Add the two acts to my first method

Drag the *act1* and *act2* methods into the **world**.*my first method Do Nothing* area. See **Figure 1-23**. {**Video demo**: Video 1-12}

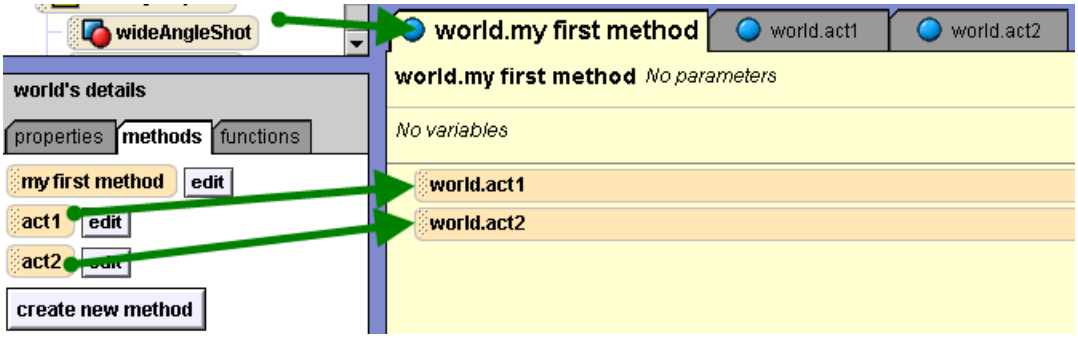


Figure 1-23 Act methods in world.my first method.

STEP 6 – Disable act2

Right-click on **world**.*act2* in **world**.*my first method* and select disable in the pop-up window. See **Figure 1-24.** {**Video demo:** Video 1-13}

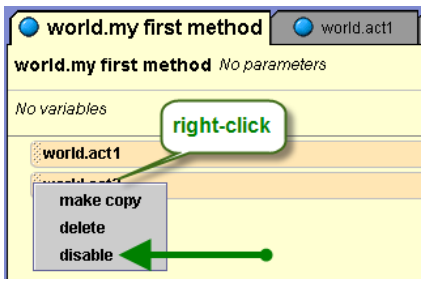


Figure 1-24 Disable a method.

For the remainder of this chapter, and for Chapter 2, we will only be dealing with the *act1* method. By disabling *act2*, only the *act1* method in **world**.*my first method* will play.

What is a user created method, anyway? They are instructions that we create in order to implement a desired behavior. In this case, by creating two act methods, owned by the **world** object, we can divide our play into easy-to-use separate containers. Each container (i.e., each act method) will hold instructions pertaining to its particular act. These act methods are executed sequentially in the order (i.e., top to bottom) they reside inside **world**.*my first method*. We can build user created methods for other objects such as **peter** and **robin** if we wish – and we will.

> *Director's comment*: To break a story into acts makes a lot of sense. After all, it's hard to keep track of all the directions, actors and scenery placements, etc. Stories and plots just seem to naturally lend themselves to being divided. Anyone who watches commercial television will recognize this as will those who attend plays.
> *Tech talk:* Breaking up a larger problem or program into parts is called "functional decomposition" which is a fancy phrase for a very simple principle of dividing a large task into smaller components (i.e., sub-tasks).

Story description details for Act 1

Let's review the content of *act1*. Wide-angle shot of the farmhouse, Robin and Peter in the grassy world. The scene is announced. Robin notices Peter out of the corner of her eye. Both characters simultaneously turn to face the camera. This is the first part of Act 1 and is as far as we get in Chapter 1.

What follows is the second part of Act 1 which we do in Chapter 2. Robin is interested in getting closer to Peter so she turns to face him and begins to move toward him 1 meter at a time. As long as she is more than 2 meters away from Peter, she continues to move toward Peter.

When Robin is within 2 meters of Peter, she stops. Peter turns to face Robin and is startled by her being so close. Peter jumps up and down twice while flapping his wings twice and at the same time spinning around twice.

Robin, herself startled by Peter's actions, backs up 1 meter and turns to face the camera. The camera moves in for a close-up of Robin's face as she blurts out the words "Oh my goodness! Sorry about that."

The scene fades to dark. End of Act 1. See **Figure 1-25** below. {**Video demo:** Video 1-2}

Figure 1-25 End of Act 1.

Creating the first part of Act 1

There are several things we need to do to begin the *act1* method. These are: move actors to their places; announce Act 1; have the actors turn to face the camera; and show Peter excited. We show you how to make these things happen.

Actors to their places

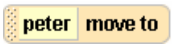
Let's figure out how to direct our actors to their initial positions. Even though we have already placed **peter** and **robin** at their respective marker cones, we did that on our own outside of the *act1* method. This was done to show you how to set invisible stage markers that designate the actors' starting points. Now we need to incorporate the actors' movements to their starting points into the *act1* method as explicit instructions. You can check out this video prior to, or as you carry out the following steps. The video demonstrates the steps below. {**Video demo:** Video 1-14}. The *act1* instructions that these steps create are shown in **Figure 1-26**.

Here are the steps to move the actors to their starting positions & set the camera to a wide-angle shot.

STEP 1 - In the **Method Editor** panel, click on the **world.***act1* tab to make it active. [If you don't see the tab there, click on **world** in the **Object Tree** panel. Click on the methods tab. Click on the edit button next to *act1*.]

STEP 2 - Click on **peter** in the **Object Tree** panel.

STEP 3 - In the **Details** panel, click on the methods tab.

STEP 4 - Scroll down and find the 

tile. [NOT the **peter**.*move* tile.]

STEP 5 - Drag and drop the tile into the *Do Nothing* area in the *act1* method. Notice that as you do this, a line will appear in the **Method Editor** panel that indicates where you are trying to place the tile. A green line indicates a permissible location.

STEP 6 - Select **peterCone** from the drop-down list. Click more…. Click *duration*. Click other. Click the 0 key. Click the Okay button. Notice that the *duration* is now set to zero. This means that **peter** will move instantaneously to **peterCone,** so fast that we will not easily see the movement.

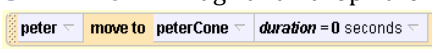
STEP 7 - Click on **robin** in the **Object Tree** panel.

STEP 8 - In the **Details** panel, click on the methods tab.

STEP 9 - Scroll down and find the 

tile.

Alice idiosyncrasy: The default duration for methods in Alice is set to 1 second.

STEP 10 - Drag and drop the tile into the *Do Nothing* area in the *act1* method under the 

instruction. Notice that as you do this, a line will appear in the **Method Editor** panel that indicates where you are trying to place the tile. A green line indicates a permissible location.

STEP 11 - Select **robinCone** from the drop-down list. Click more…. Click *duration*. Select 0 if you see it. If you don't see it, click other. Click the 0 key. Click the Okay button. Notice that the *duration* is now set to zero. This means that **robin** will move instantaneously to **robinCone**, so fast that we will not easily see the movement.

The resulting *act1* method so far is shown in **Figure 1-26**.

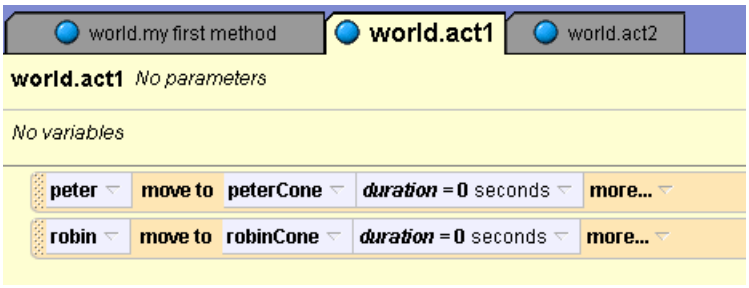


Figure 1-26 Actors to their positions.

STEP 12 - Set **camera** to **wideAngleShot**. With the *act1* method tab open in the **Method Editor** panel, click on **camera** in the **Object Tree** panel.

STEP 13 - In the **Details** panel, click on the methods tab. Then, drag and drop the 

tile under the 

instruction.

STEP 14 - Select Dummy Objects/**wideAngleShot** from the drop-down list. The resulting instruction should be:

Before you click the Play button, move **peter** and **robin** close together in the **World View** panel. Then, click the Play button to observe what you created. They should both move quickly to their cones. Close the World running … window when done.

Announce act1

Now that we have instructions for moving our actors to their starting places, let's add instructions to begin our movie by having the **farmHouse** object announce the scene by displaying the text "Hello" (or whatever you like).

STEP 15 - Click on the **farmHouse** object in the **Object Tree** panel and notice the **farmHouse's Details** panel. Be sure that the methods tab is active. The fifth method down is the **farmHouse** *say* method. Drag and drop the **farmhouse** *say* tile to the **Method Editor** panel under 

in the *act1* method. A line will appear in the **Method Editor** panel that indicates where you are trying to place the tile. A green line indicates a permissible location. In the *What* drop-down list that opens, choose other**…**, and in the Enter a String window that opens, type "Hello." Click on OK. {**Video demo:** Video 1-15}.

Then, click on Play to see the results. Ooops! That was pretty quick. The "Hello" needs to display longer. Close the World running … window.

In the instruciton containing the *say* message to **farmHouse** click the down arrow next to more… and choose *duration/*2 seconds. See **Figure 1-27.** Click Play again and notice the longer duration. {**Video demo:** Video 1-16}

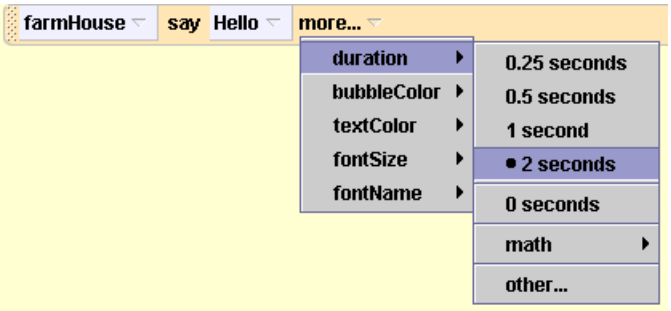


Figure 1-27 Setting the duration for farmHouse say method.

Actors turn to face the camera

From their starting positions, **robin** and **peter** need to turn to face the **camera**. See the resulting directions in **Figure 1-28.**

STEP 16 - Tell **peter** to face the **camera**. Click on **peter** in the **Object Tree** panel. In the **peter's Details** panel, be sure that methods tab is active. Scroll down until you see the method **peter** *turn to face.* Drag and drop it under the **farmHouse** *say* instruction in the *act1* **Methods Editor** panel. In the target drop-down list, select **camera**.

STEP 17 - Tell **robin** to face the **camera**. Click on **robin** in the **Object Tree** panel. In the **robin's Details** panel, be sure that methods tab is active. Scroll down until you see the method **robin** *turn to face.* Drag and drop it under the **peter** *turn to face* **camera** instruction in the *act1* **Method Editor** panel. In the target pop-up window, select **camera**. {**Video demo**: Video 1-17}.

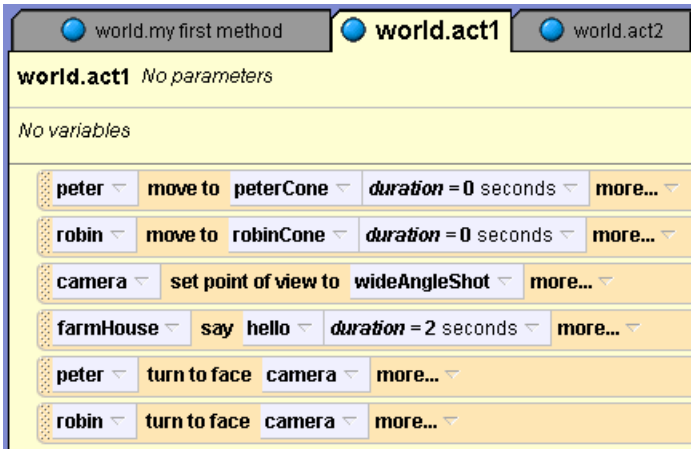


Figure 1-28. Insturctions to peter and robin to face the camera.

Click the Play button in the upper left-hand corner of the screen. Note that **peter** turns to face the **camera** first, and then **robin** turns to face the **camera**. Ooops! This is not what we want our actors to do. We want them to turn together.

STEP 18 - To get these two actions to occur simultaneously find the Do together control at the very bottom of the *act1* **Method Editor** panel. It is in the row of controls that looks like this:

Drag and drop the Do together control between the instruction to the **farmHouse** object and the instruction to **peter**. See **Figure 1-29.**

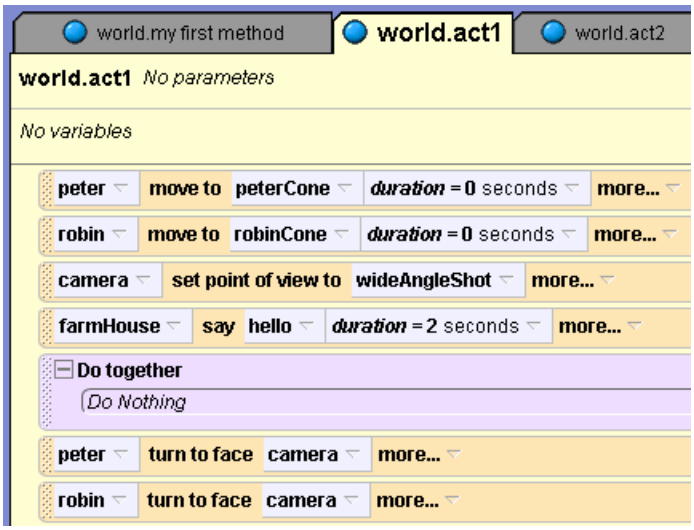


Figure 1-29. Do together control.

STEP 19 - Now, in the *act1* **Method Editor** panel, drag and drop the **peter** turn to face **camera** instruction into the *Do Nothing* part of the Do together control. Drag and drop the **robin** *turn to face* **camera** instruction under the **peter** *turn to face* **camera** instruction in the Do together control. Actually, their order in the Do together control does not matter. See **Figure 1-30**. {**Video demo**: Video 1-18}.

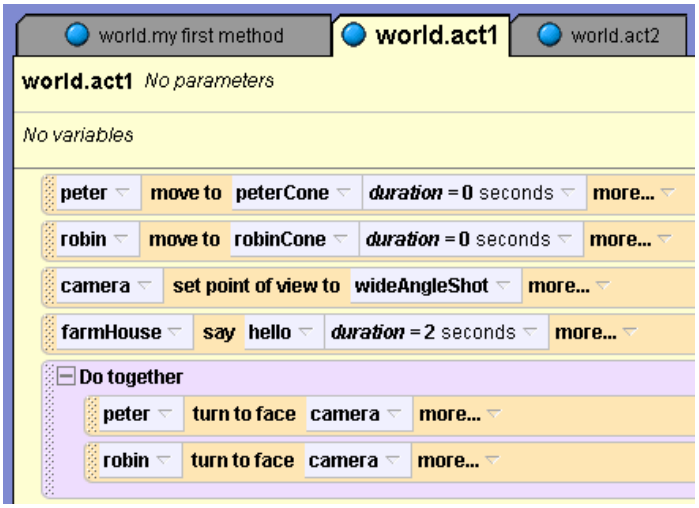


Figure 1-30 Do together control containing peter and robin instructions.

Click Play to see the results. That's more like it! Both characters turn to face the camera at the same time.
Suggestion: This might be good time to again save your world

Special Note about Terminology: Objects - Methods - Messages - Parameters

Each of the lines you see in Figure 1-30, is called an Alice program statement (aka, instruction). The general format of a statement is the name of the object that is to receive a message followed by the message being sent to it, and sometimes followed by more message information (called a parameter).

object message <parameter>

For example, the 3rd instruction you see in Figure 1-30 is shown in the picture below.

This instruction consists of the message set point of view to that is sent to the camera object. When an object (the camera in this example) receives the message, it implements a METHOD it knows that has the SAME NAME as the MESSAGE it received. [Emphasis is intended!] This produces the object's behavior in response to the message.

As an analogy, suppose you sent a text message to one of your friends  asking him or her to meet you at the movies. Your friend  will then implement the behavior needed to get him or her to your meeting place. Note that you do not tell your friend how to get there. Implementing the required behavior is his or her responsibility and you do not need to know how he or she carries out the desired behavior – only that he or she can.

Sometimes it is necessary (or desirable) to send information along with a message. For example, in the 4th instruction you see in Figure 1-30 is shown in the picture below.

In addition to sending the say message to the farmHouse object, we are also sending along the text "Hello" and the duration of 2 seconds. This information, known as parameters, tells the farmHouse object what to say and how long to say it

Continuing with the analogy of sending a message to a friend, suppose that you also sent the information "place" and 'time' along with your request to meet. "Place" and "time" are parameters attached to your message to your friend. For example: john meet-me strand-theater 8pm Tuesday.

There is one additional rule about sending messages to objects and that is you can only ask objects to do things they know how to do (i.e., the methods they own).

Peter gets excited

Note: We are going slightly off-script here just to get some action going before we move to Chapter 2 where we will finish up *act1.*

Here is what is going to happen in our off-script journey. Still in the wide-angle camera view, **peter** gets excited and flaps his wings two times as he falls backwards on the grass making a clucking sound and a splashing sound. We realize that penguins don't cluck like chickens nor do they typically make a splash sound effect when falling on the grass. But, in a virtual world, you can pretty much do what you want - even defy the laws of gravity!

Steps 20 – 26 are shown in: {**Video demo**: Video 1-19}

STEP 20 - Drag another Do together control into the *act1* in the **Method Editor** panel and place it under the already existing Do together control. See **Figure 1-31.**

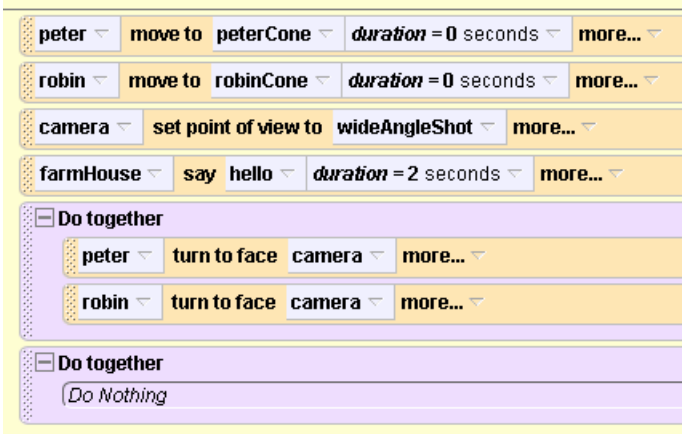


Figure 1-31 Do together control for peter's excitement.

STEP 21 - In the **Object Tree** panel, click on **peter**. See **Figure 1-32**. Notice that the **Details** panel shows **peter's** methods. If this does not show, just click the methods tab. One of peter's methods is *wing_flap*. Another method is called *turn*. And another is called *play sound* (you may have to scroll down to see this one).

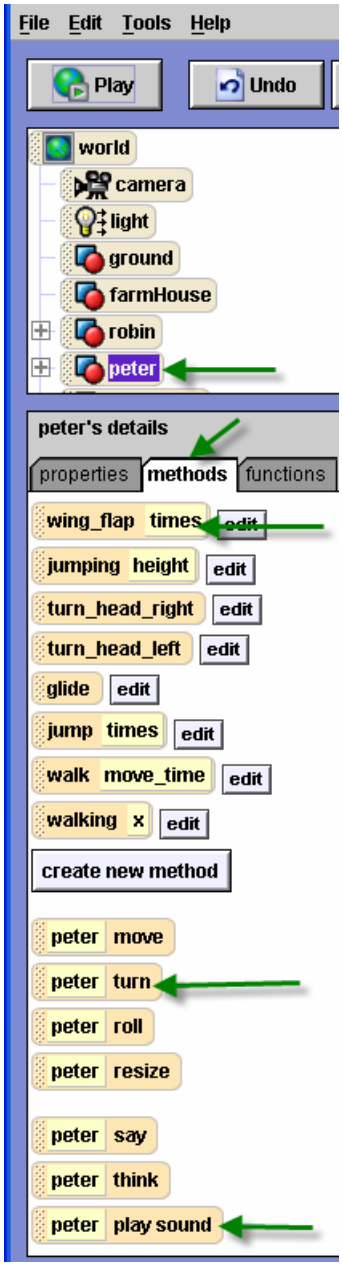


Figure 1-32 peter's methods.

The results of the steps below are shown in Figure 1-33.

STEP 22 -Drag the *wing_flap* method into the Do Nothing part of the second Do together control in the *act1* **Method Editor** panel and select 2 when prompted.

STEP 23 - Drag the **peter**.*turn* method into the Do together control in the *act1* **Method Editor** panel and place it below the **peter**.*wing_flap times*=2 method. Select backward and then 1/4 revolution when prompted.

STEP 24 - Drag the *play sound* method into the Do together control in the *act1* **Method Editor** panel and place it below the **peter**.*turn backwards*.25 revolutions method. Select chicken. Click on more… and chose *duration* 2 seconds.

STEP 25 - Again, drag the *play sound* method into the Do together control in the *act1* **Method Editor** panel and place it below the **peter**.*play sound world. chicken* method. This time select splash. Click on more… and chose *duration* 2 seconds.

See the complete *act1* method in **Figure 1-33.**

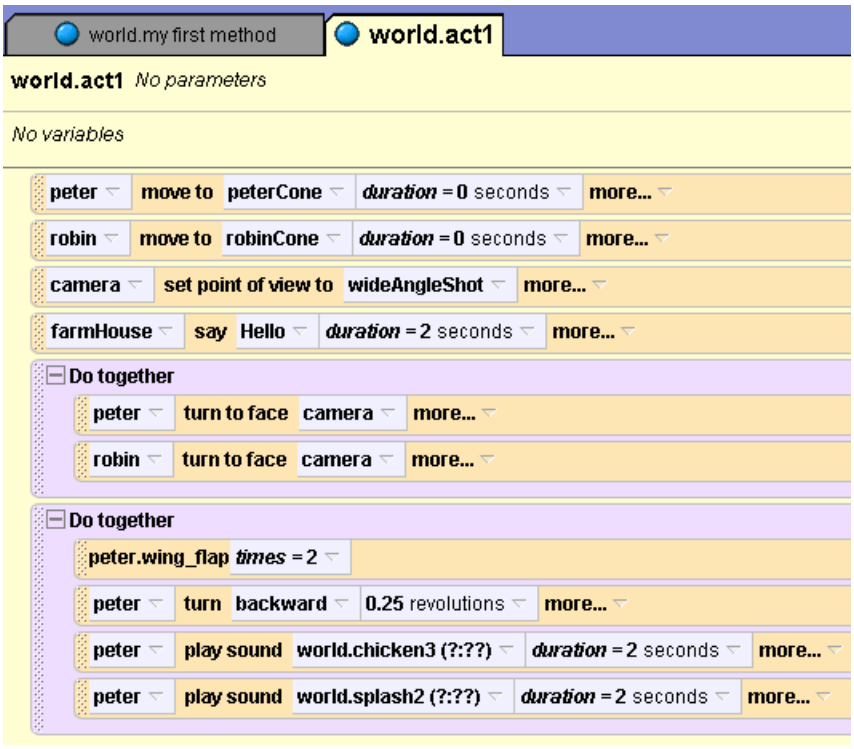


Figure 1-33 act1 method.

With your computer sound on click the Play button. The final animation can be seen in this video demo: {**Video demo:** Video 1-1}. Good to save your work now. Also, you can create a video of your animation for later viewing and/or posting to a web site. See Appendix A - Section A.3: Exporting your movies for QuickTime Videos & WebPages.

Final words on Chapter 1

We end Chapter 1 here but will pick up on *act1* again in Chapter 2. We went off script and added something extra to the story: **peter's** wing flapping, falling backwards, and some sound effects. This illustrated how easy it is to add instructions for our actors and demonstrated additional things that objects can do. At the start of Chapter 2, we will just as easily remove them since they are extraneous to our original story. Directors are always fiddling with movie, play and game scripts.

Try the Chapter 1 Exercises that follow. They are meant to give you practice on what you already know how to do and to also stretch your knowledge and skills. Of course, you are encouraged to explore Alice on your own. You can't break anything, so have fun! You can always save your worlds as different file names by using File/Save World As… as you experiment.

Also, read the Chapter 1 Skills, Concepts, and Capabilities section to find out more about the theory and competencies you have learned and how they relate to the *Fluency with Information Technology* text book.

Your authors' version of the end-of-Chapter-1 Alice file is posted to a web site should you wish to refer to it at a later time. For example, you might want to start off your Chapter 2 work with it.

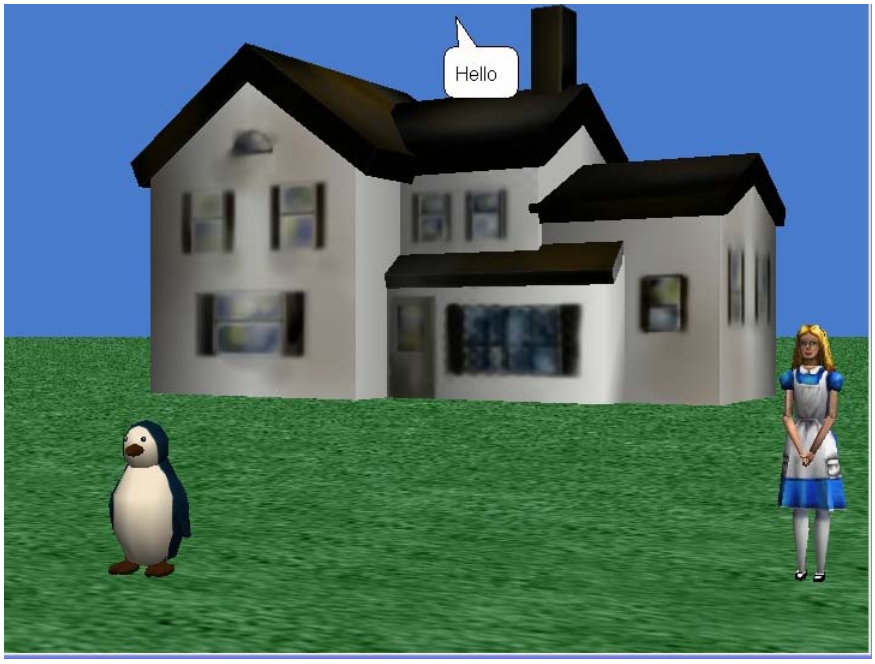
Authors' file AWB1-1.a2w to this point in the chapter can be found at

http://media.pearsoncmg.com/aw/aw_snyder_fluency_3/alice/World_Files.

Chapter 1 - Exercises

In Chapter 2 we'll complete Act 1 of our movie. For now, let's play around with Alice and get comfortable with it. Don't worry about making mistakes, and remember that you can always Undo anything that goes wrong. Try the following exercises.

1. In the *act1* instruction "farmHouse say Hello -- *duration* 2 seconds" you'll see several downward arrows that allow you to select other values. Fool around with selecting other values. Play the world, and see what you get.
2. Do the same for the instructions that have Peter and Robin turning to face the camera.
3. Add one other command in the Do Together control that makes the Farmhouse do something crazy, *move* up and down for example.
4. Instead of Peter falling backwards at the end of the chapter, have him rise off the ground by 2 meters and roll around. You can use the *move* and *roll* methods.
5. Add a few other objects to the scene and have them do something interesting and/or funny.



Chapter 1 – Skills, Concepts and Capabilities

This is not the kind of end-of-chapter summary you may be used to seeing in textbooks. Here we tell you how what you have learned in the chapter relates to Snyder *Fluency with Information Technology* textbook concepts, skills and capabilities.

Our Alice Fluency Workbook Chapter 1, by its nature, was introductory. In subsequent chapters, these *Fluency* skills, concepts, and capabilities will be reinforced and you will master new ones.

While you've been having fun making animated movies with Alice, you've been learning some important things about algorithms and computer programming.

With regard to the five essential properties of algorithms, we specified what the play or movie would look like up to this point (output), specified directions for each stage of the movie (definiteness), knew that Alice can carry out your directions (effectiveness) and that the movie stopped when completed (finiteness). When you specified number of wing flaps (i.e., 2), you were providing input data to the program/algorithm (input property of algorithms).

Also, you've learned that computers execute instructions (i.e., directions) sequentially, unless told otherwise (e.g., the Do together control over-rode sequential execution).

You learned that in Alice (and all object-oriented programming environments) the fundamental building blocks are **objects** which have their own methods, properties and functions. You will meet functions in Chapter 2.

Alice Tutorial: Try the Alice Tutorial 1 to reinforce some of the things that you learned to do in this chapter.
If you are in Alice, click on File and then on New World. You will see the Welcome to Alice! screen. If you are already at that screen, just click the Tutorial tab and click Tutorial 1. Then click

Using the Alice graphical user interface (GUI), you were able to create a set of instructions that had objects (i.e., **peter, robin** and **farmHouse**) do something when the starting event triggered program execution (When the world starts …).

You learned that in Alice, names stand for things such as objects (e.g., **farmHouse**) and that multiple instances created from classes found in the Gallery become independently named objects (e.g., **peterCone** and **robinCone**).

You learned that: an event is triggered when the Play button is clicked and **world**.*my first method* is launched via When the world starts; that a movie or play script can be broken down into parts which can contain directions (i.e., instructions); and that those directions, invoked in the right sequence, can create an Alice animated computer movie.