Creating Functions

By Deborah Nelson
under the direction of Professor Susan Rodger
Duke University
July 2008
www.cs.duke.edu/csed/alice/aliceInSchools

Introduction and Set Up

Starting from an existing world, this tutorial will help you to make the existing methods smarter by asking questions about the objects in the world.

Download the starting world that goes along with this tutorial from the website:
http://www.cs.duke.edu/csed/alice/aliceInSchools

It will be named fireStart.a2w

Save the file in a directory that you can find later.

Start Alice and open the world

Note: You cannot double-click the file to open it. Windows will not know what to use, and even if you select Alice from a list of programs, the loading will fail.

Introduction and Set Up

In this world, the user is supposed to burn anything that is in the junkyard.

Play the world.

There is an error because, whatever the user clicks on burns – even the animals and the ground.

We only want the things in the junkyard to be burned. So we need to write a function.

Part 1: Functions

Introduction

Information about the world or the objects in the world are all stored in properties.

A function is used to ask questions about these properties.

For example, some of the built in functions of an object are: color, height and depth.

A function is not a behavior or an action. It simply returns the information that we need.
Part 1: Functions
Introduction Continued...

In this world, we want to know: is the object that was clicked on one of the objects in the junkyard?

If it is one of the objects in the junkyard, we want the fire to go to it.

If it isn't, we don't want anything to happen.

Part 1: Functions
Types of Functions

Depending on the question, a function can return any type of value.

Some of the options of a return type are:
• Number
• Boolean (true or false)
• Object
• String

But there are more types

Part 2: Creating a Function
Step 1: What type of function to create

We want our function to be type boolean. Boolean returns either true or false.

This function will return true if the object clicked on is in the junkyard. It will return false otherwise.

Part 2: Creating a Function
Step 2: How to Create a Function

Click on the function tab in the world detail's

Click on create new function

Name it possibleObject and make sure it is type Boolean
The object that was clicked on is going to be used in this function.

In the possibleObject function click on create new parameter
Name it possibleFuel with type Object.

Drag the parameter possibleFuel on top of the true

Select possibleFuel ==

select cardboardBox and then the entire cardboardBox

Drag an If/Else into your function and select true in the drop down menu.

Drag return on top of the first Do Nothing

Select true

This means if the object that has been selected is the cardboardBox then the function answers true, it can be burned. If it is not, “do nothing”, for now.
Part 3: Writing the Function
Step 5: Nested If Statements

We are now going to check each of the burnable objects by using several if statements.

Drag If/Else on top of the Do Nothing under Else

Then drag possibleFuel on top of the true

select possibleFuel==
select crate
select the entire crate

Part 3: Writing the Function
Step 6: Finish the function

Drag a new If/Else onto the Do Nothing under Else

Repeat the steps for these objects:
• The teddyBear
• The table
• The oldBed

Then we want to return false for any object not in the junkyard

Your function should look like the screenshot on the next slide.

The Rest of the function

Don't forget to change the final return to false
Part 4: Calling the Function

Click on the methods tab in the world details pane.

Click edit beside the method lite so that you can see the code:

```plaintext
G#62,#576
(1#-75#(62#(G#62,5(#576(-,(62$6(/,*(+$&(-##(62#(+,5#(
```

Click on the functions tab in the world details pane.

Part 4: Calling the Function Continued...

Now drag the possibleObject from the function tab on top of the true.

Select expressions
Select objClick

Part 4: The If

Click on the already written Do in Order

Drag it into the If/Else statement on top of the Do Nothing

Play your world
Part 4: The Else

Scroll down to the Else and drag the camera say method and type in a string “that object shouldn't be burned”:

```
Else

camera say that object shouldn't be burned more...
```

Part 5: One More Error

Now when you play your world, only objects that are in the junkyard are burned.

However, when you burn one of the objects, you can still burn it again.

Let's nest another if statement in the lite method to check whether or not the object has been burned.

To know if something has been burned we can check it the color is black.

Part 5: One More Error

Step 1: Writing a New Function: Is it Burned?

Now, click on the function tab in the world details pane.

Create a new function named notBurned with the type Boolean

Drag an If/Else into the function.

```
world.notBurned. No parameters

<table>
<thead>
<tr>
<th>true</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do Nothing</td>
<td>Do Nothing</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Return true
```

Part 5: One More Error

Step 1: Writing a New Function: Is it Burned?

In this function, we want to return true if the object has not been burned. We know an object has been burned if it's black.

Click on cardboardBox in the object tree

Click on it's properties tab

Drag color on top of the true in this function

Select != black
Part 5: One More Error
Step 2: Return Statements

Now drag `return` on top of the first **Do nothing**.
Select **true**.

Drag `return` on top of the second **Do nothing**.
Select **false**

Select **false** for the last `return`.

---

Part 5: One More Error
Step 3: Create New Parameter

Click on `create new parameter` in the function

Name it `objClicked` with type **Object**

Drag `objClicked` on top of `cardboardBox` in the function's `If` statement.

---

Part 5: One More Error
Step 4: Nest an If/Else

Click on the `world.lite` tab to see the code

Drag in a new `If/Else` statement

For the Else of the statement, drag in the `camera say` method and type in: “**that object has already been burned**”

---

Part 5: One More Error
Step 4: Nest an If/Else Continued...

Now click on the already written **Do in order** and drag it on top of the **Do Nothing**
Part 6: Calling the function notBurned

Click on `world` in the object tree.
Click on the `functions` tab
Drag the function `notBurned` on top of the true in your if statement.

Then drag the parameter `objClick` onto the `<None>`

Play your world

Recap

A function is not a behavior or action
It is used to return information about the state of the world or the characters in the world

There are many different types of functions. For example, functions can return objects, numbers, booleans, etc.

Write your own functions based on questions you need to answer about your world.