Shorter of two objects and changing color

Functions, events and setting the color

Susan Rodger, Duke University
June 2008
What this demo shows...

• You’ve used built-in functions. Now you will write your own functions.

• Part 1: Create a “world” function that returns the shorter of two objects – it is a world function because it has two different objects.

• Part 2: Create a chicken function that changes the color of the chicken (cycling through 4 colors).

• Part 3: Create a snowman function to change the color of the snowman randomly.
Notes

• Not all objects can change color. The chicken and snowman can change color, which is why we use them. The bunny cannot change color, the object was just designed that way.
Part 1: Start Alice world and add in chicken, snowman, bunny, and joey
Create World function

“ObjectThatIsShorter” – select Type as Object
The new function appears – notice the “Obj” – means this function returns an object
Compare the chicken and snowman’s heights – drag up the if, from world functions, drag over the “a<b” and enter 1’s.
Now click on Chicken, then functions, and drag Chicken’s height over twice
1. Click on “create new parameter” and create “object1” with type object. Do again with “object2”.
2. Then drag them over the word chicken.
Drag up the “Return” twice to replace the “Do nothing”s
Then drag in object1 and object2, returning the shorter object
Now try out the function, use it where there is an object.

In myFirstMethod, put this code...
Replace “chicken” with new function – in World Functions, drag over the function “objectThatIsShorter” (that returns an object) over Chicken. Then Play

objectThatIsShorter can be used in place of any object.
Call the function several times to compare heights of different animals, then press “play”
Resize objects to make some taller, and click Play again
Part 2: Change the color of the chicken from “no color” to “blue” to “red” to “green” and cycle through again
Create Chicken Function “changeColor”
Click on Chicken, then functions, then “create new function”, type “changeColor” as name, and type “Color”
The new function appears – notice the color wheel – this function returns a new color for the chicken.
Returning new Chicken color - Idea

- If the chicken color is “no color” then we want to return the new color “blue”
- Else, if the chicken color is “blue”, then we want to return the new color “red”
- Else, if the chicken color is “red”, then we want to return the new color “green”
- Else if the chicken color is “green”, then we want to return the new color “no color”
First Drag up the if/else. Then select “Chicken”, “properties”. Then drag over the color property and select “Chicken.color == no color”
The “no color” when dropped looks like white,
Now drag up “return”, drop in after the if, and select “Blue”
Under the “else” part, drag in another if/else, and repeat the second if (if the color is blue, return red)
Now continue with the other two cases.
If red return green
If green return no color
Here is the final code... with four nested if’s
Now, let’s use the new function Chicken.changeColor. When we click on a Chicken we will change its color to the new color returned from this function. First create an event.

```
Events
---
create new event

When

When the world starts
When a key is typed

When the mouse is clicked on something
When something is true
When a variable changes
Let the mouse move <objects>
Let the arrow keys move <subject>
Let the mouse move the camera
Let the mouse orient the camera
```

```
Events
---
create new event

When the world starts, do

world.my first method

When is clicked on anything, do
Nothing
```
First, select Chicken to click on, then drag over color property and under expressions, pick the color returned by the function chicken.changeColor
Now click Play and then click on the chicken several times.
Part 3: Click on Snowman, functions, then create new function, with name `changeColorRandom` and type `color`
How to change color randomly

- Generate a random number in Alice between 0 and 1
- If the number is between 0 and .25, return color “red”
- Else, if the number is between .25 and .50, return “blue”
- Else if the number is between .50 and .75, return “green”
- Else if the number is between .75 and 1, return “no color”
First, create a “local variable” to store the random number, this is like a “property” for the function only, make sure to select Type: number.
1. The local variable appears
2. Drag it down to the code and set its value to 1.
Select “world”, then “functions” to find “random number”, and drag it over
Click on “more” to add minimum set to 0 and maximum set to 1. randomNumber stores the number generated so we can refer to it.
Drag up an if/else
Check randomNumber for red case: If randomNumber > 0 and < 0.25. From “world”, “functions”, drag over a>b, then fill in
Click on the last white down arrow, select “logic”, ... “and”, and “true”
Drag in the “a<b” and then fill in values.
Drag up the return, and select the color red
Drag another if/else into the else, and drag in code for the “blue” case.
Add “green” and “no color” cases
Create and event for clicking on snowman, then drag snowman’s color property over.
Click “Play” and click on the snowman and he randomly selects one of the four colors.
How do you know if it is working correctly? Add a print to see the randomNumber value. Select Object, and ground.
Drag randomNumber over “ground”. Click on down white arrow and type in text: This is a randomNumber
Now play, click on snowman and you can see the random Numbers.