Moving Across Automatically

(10 pts, due next Thursday)

Contents

Intro

In the last tutorial, you saw how to use the absolute positioning and the .style to reposition an element on your web page by clicking. You also had global variables that were used by multiple functions to reposition the element.

We're going to extend that in this Tutorial. We're going to have html elements move independently using setTimeout.

While we're combining and using these tools to make a game, all of the elements can be used in combination in an infinite number of ways. One simple example is using the tools we're learning today is a banner that scrolls across a screen. My field is assistive technology, or technology for people with disabilities. A couple of ways in which the tools we've learned are used in my field alone include:

- A Training tool in which random pictures are shown and users are asked to identify something about the picture (e.g.,emotional expressions for individuals on the autism spectrum) and gets points for correctly identifying the pictures
- A training coordination tool in which a dot/image appears in a random place on the screen and users have to click on or near the image/dot
- A dynamic interface that allows the user to personalize the appearance of their screen based on their particular needs (large font for low vision, black background with white text for high contrast, etc.)
- A user interface for a prompting system that helps to give visual and text cues to individuals with aphasia (head trauma)
- Many, many more!

The game we're creating involves having a hero gather goodies around the screen while avoiding a bad thing, and the hero wins if they gather 10 good things before getting hit by the bad thing. To make it a simple training tool, you could modify it by having a multiplication problem appear at the top of the screen, and then making the hero gather an image of the correct number that would be the answer. Or, a word in text form could appear at the top of the web page and the hero must gather the image that represents the text (as a reading teaching tool). I'm just throwing out possible modifications to turn even the game we're working on into an assistive tool.

In my opinion, one of the coolest aspects of computer science is that once you've mastered the tools, you can create just about anything you can imagine. The challenge is figuring out how to take the tools you have to create what you want to create.

Back to the tutorial:

For this tutorial, you're going to be picking the bad element in your game and making it move automatically. Mine is an asteroid that can come down and hit my hero and destroy him. Your bad element could be a hunter in a jungle scenario, a shark in the ocean, a rabid snail in a garden scenario – whatever you want.

Step 1: Add an image to your web page. The image should be of the thing that will be your bad thing in your game. Mine was an asteroid, although I seriously thought about making it be a clown.

Step 2: Make sure you position the image absolutely. Wherever you position the image will be where it starts its movement. In my case, I positioned mine off the page on the top, so I actually positioned it -200 from the top (if the top is 0, every pixel lower on the page is a higher number, which is kind of counterintuitive. So the bottom of the page might be about 600 pixels down from the top.)

Step 3: Add a button to your html code. Make the value be "Start movement" and make the onClick call a function MoveBad with the id of the image (above).

Step 4: Save your code

Step 5: Now in your .js javascript file, add two global variables at the top for positioning your bad thing (just like you did for moving your hero). Make sure you name these two global variables something different than you named your global variables used for positioning your hero. I named mine btop and bleft for bad thing's top and bad thing's left positions.

Step 6: Set the bad thing's top and left position to be exactly to what you set them to be in your html code. So if you set the top to be -200 in your html code, then btop=-200. Same with the bleft (this is exactly the same as what we did with the hero).

Step 7: Create a function MoveBad(par).

Step 8: In the Movebad, add 10 to the btop variable (again, just like you did in the moveDown function for your hero).

Step 9: using document.getElementById(par), set the position to be absolute (again, just like the move functions for your hero).

Step 10: use document.getElementById and the .style.top to set the top position btop+"px" down from the top .

Step 11: Test by saving and clicking on the start button. The bad thing should move down the page 10 more pixels every time you click.

Moving Automatically using setTimeout

So far this is almost identical to Move functions you wrote for your hero. But what we want is for the bad thing to move automatically without our having to click. To do this, we'll add setTimeout to the bottom of the function:

Step 12: Add a setTimeout to this function so that it keeps moving automatically. The line should be the last line in the function (right above the closing squiggly bracket) and should look as follows:

```
setTimeout(function(){MoveBad(par);},500)
```

Step 13: Save and test. You should be able to click on the button that called MoveBad and watch the image move automatically down the page. If you want it to move faster, change the pause time from 500 to a smaller number. If you want it to move slower, change the 500 to a larger number.

Cool, huh!

Repositioning back to top of page:

Right now the bad thing just moves down the page forever and ever. What you probably want is for it to go to what you decide is the bottom of your page, and then go back to the top and start over again. To do that, you'll add an if condition.

Step 14: Add an if condition to your function.

The if condition will work as follows: if the bad thing's position from the top (which is held in the btop variable) is greater than the location at the bottom of the screen (maybe 600, although you can play with the number), set the btop to be 0 (or -200, or a number that would put the position should be set back to 0 so it starts over at the top.

Step 15: make sure you add to the if condition the setting of btop to 0 (or -200)

Step 16: Save and test. Now when the bad thing gets to the bottom of the page, it should go back to the top of the page and start over.

Randomly moving over from left:

We're getting there, but I really would rather the bad thing moved around randomly when it starts over at the top of the screen so it's not always falling from the exact same place. To do that, when it gets to the bottom, I'm not only going to reset the btop position to be back to the top, I'm going to generate a random number for the number of pixels over from the left, set the bleft to be that random number, and then use document.getElementById to uses style.left to position the bad thing over from the left that random number (now in bleft).

Step 17: right below where you set the btop to be 0 (or -200) set bleft to be a random number between 0 and maybe 600 (again, you get to decide the size of the area you want your game to occupy.

The 600 is the farthest over from the left that the asteroid can be positioned, so the position will be somewhere between 0, which is the left edge, and 600, which will be the right side.

Step 18: use document.getElementById(par).style.left to set the left to the random number (held by bleft) right after you generate the random number and put it in bleft (Step 17).

Step 19: Save and test. Your bad thing should move down the screen, and when it gets 600 pixels from the top, it should move back to the top and some different, random location over from the left and start over.

Your Turn:

- (10 pts) Get the above working.

Extras:

There are so many variants:

- you could make the speed vary by having a variable to hold a random number between 25 and 500, and then in the setTimeout function, use that variable for the speed
- Instead of having the bad thing move 10 pixels each time, you could generate a random number between 5 and 40 and have the bad thing move down that random number each time.
- You could have it move both down and left (or down and right) by adding to both the btop and the bleft each time.

That's just a few ideas that could make the movement more interesting.