Final Project--The Amazing Box Generator!

Lesson Objectives

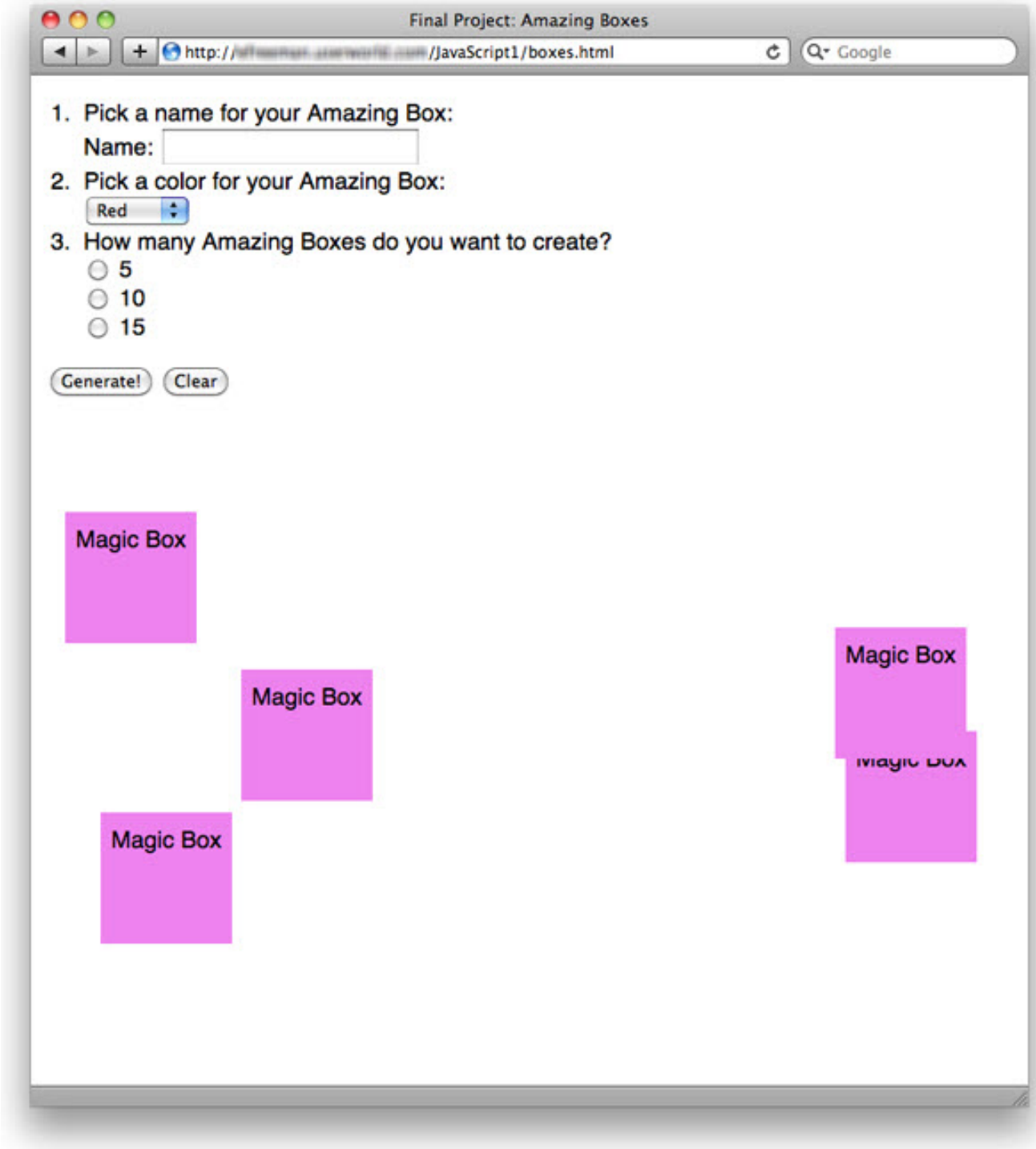
When you complete this course, you will be able to:

- create a web application called "Amazing Boxes."
- get users to select color, number of boxes to generate, and name the boxes.
- create a menu of colors in a form.
- create a box object with properties.
- add divs that represent boxes to DOM.
- style the divs.
- generate a button.
- clear the button and remove all divs.
-

You've come a long way in this course, from not knowing any JavaScript, to knowing quite a lot! Congrats. This last lesson is about the final project, which is to create a web application called "Amazing Boxes." What are amazing boxes, you might ask? Read on to find out...

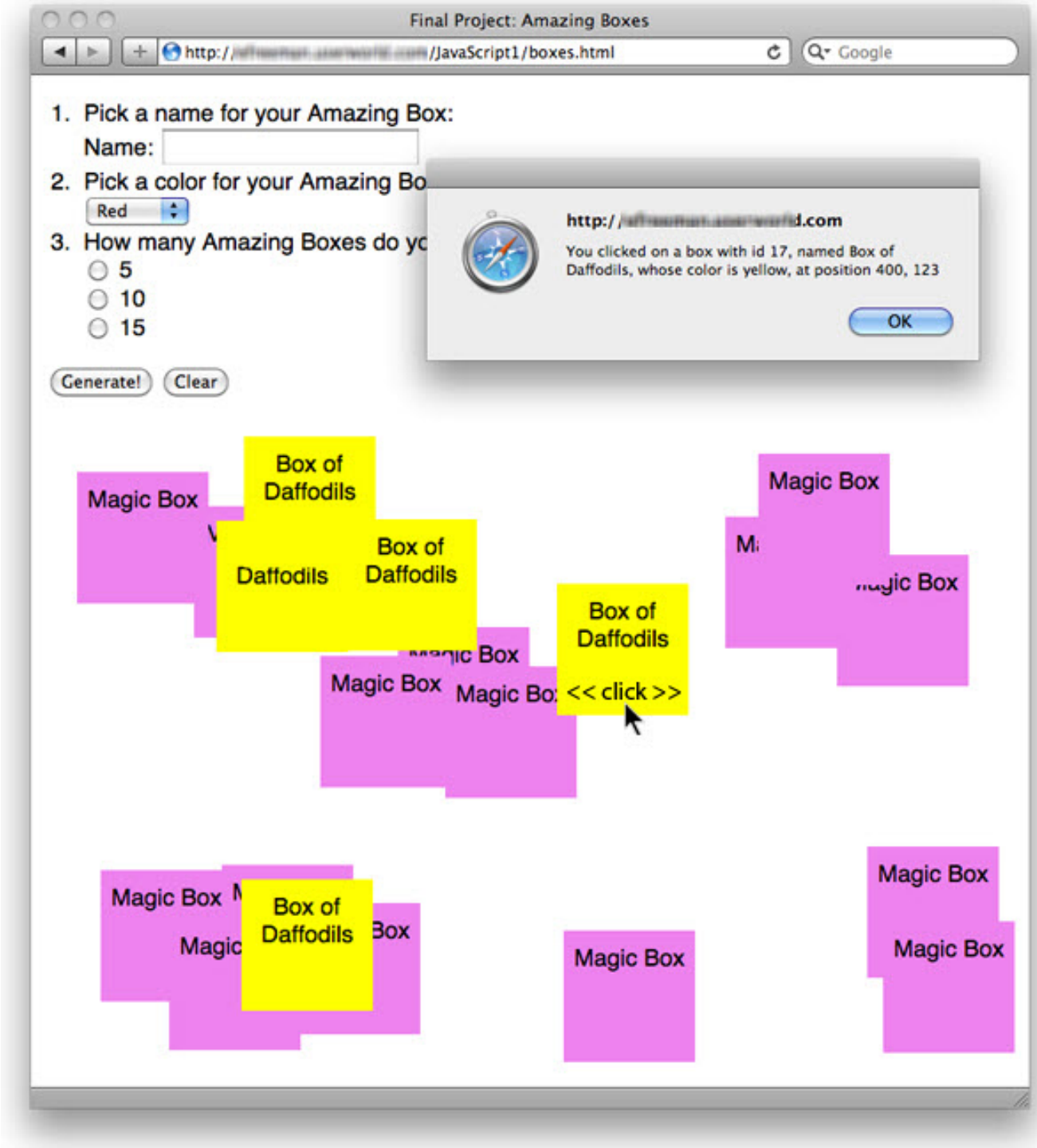
Amazing Boxes

This app has a web page with a form that prompts you for three things: the name for your amazing boxes, a color for your boxes, and how many you want to generate. Let's say you enter "Magic Box" for the name, you choose "Violet" for the color, and you select 5 as the number of boxes to create. Here's what you'll see when you click the **Generate!** button:



Five boxes (implemented as elements) are added to the page, with "violet" as their background color, the name of the box as the content of the box, and their positions randomly selected inside the part of the page where they have been added.

You can go ahead and generate more amazing boxes (no need to reload the page); you could then enter "Box of Daffodils" for the name, choose "Yellow" for the color, and choose 5 again for the number. When you click **Generate!** you'll see:

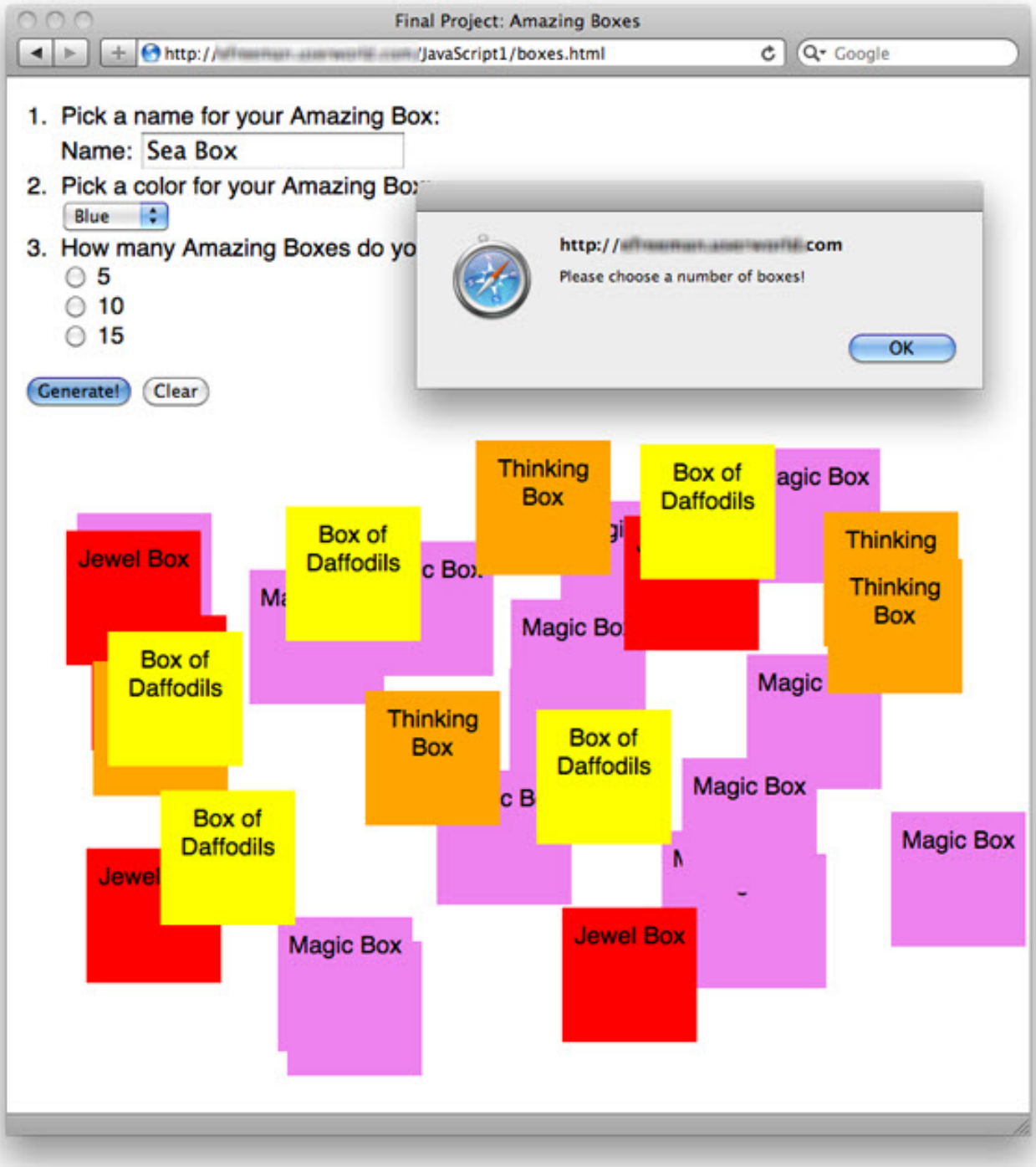


Notice that the new boxes are *added* to the page and that the "Magic Boxes" are still there. Also notice that if you click on one of the boxes, you see information about the box: its id (a unique number), name, color, and position in x, y (left, top) pixels, in the page.

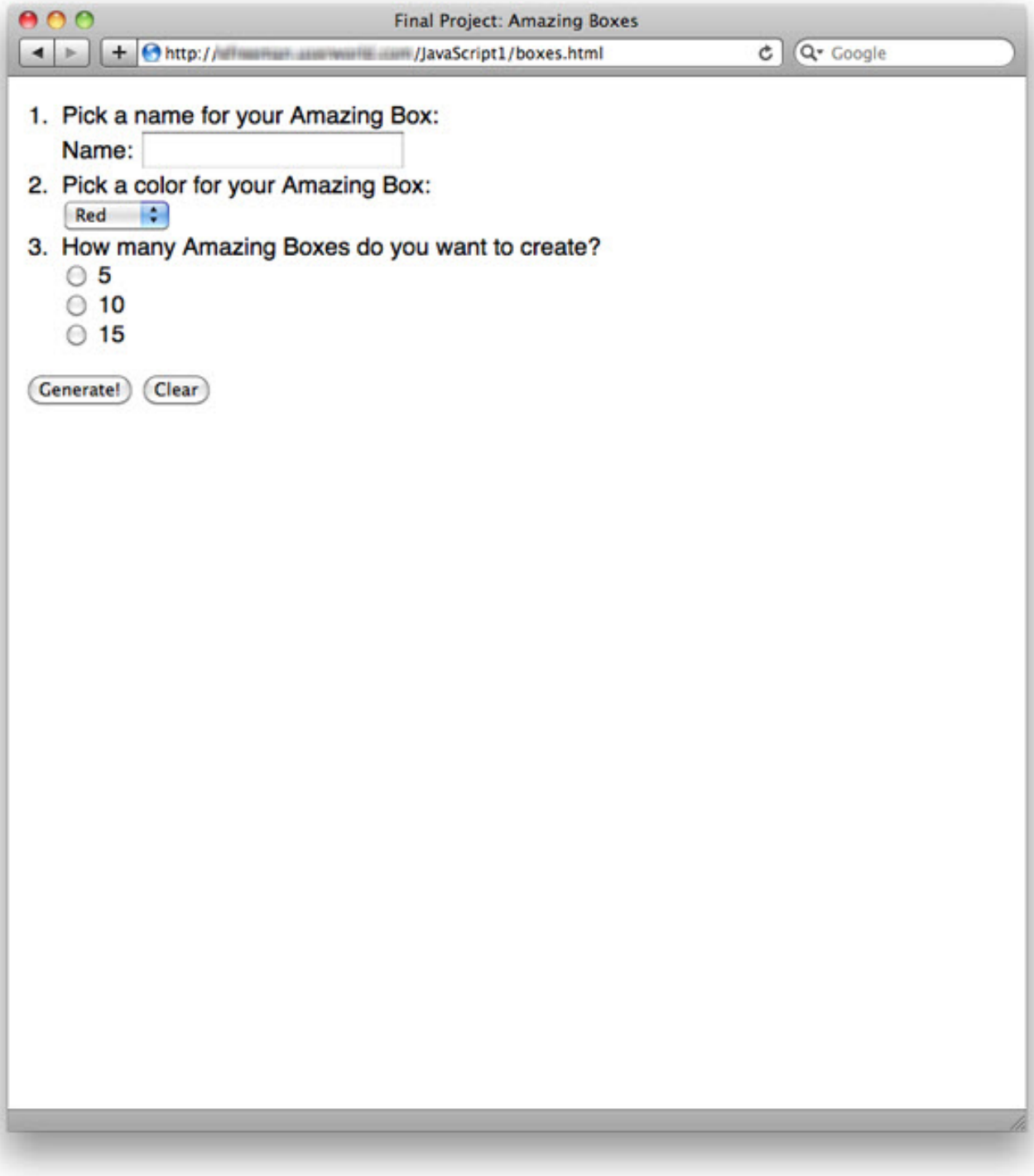
You can keep adding more boxes, as many as you like:

A nice feature to add to the app is to reset the form after clicking the **Generate!** button (like we did in an earlier lesson) so it's easier for the user to enter the next set of values.

If you forget to enter a name or choose a number, you will see an alert indicating you need to enter the value:



Finally, if you click the **Clear** button, all the boxes will be erased! You can start over and add new boxes at that point if you want.



We'll give you the HTML and CSS for this project (aren't we nice?!), and it's your job to write the JavaScript. You already have just about everything that you need; we'll fill in a couple of pieces that will help after you take a look at the HTML and the CSS.

The HTML for Amazing Boxes

Here is the HTML for the application:

CODE TO TYPE:

```
<!doctype html>
<html lang="en">
<head>
  <title> Final Project: Amazing Boxes </title>
  <meta charset="utf-8">
  <link rel="stylesheet" href="boxes.css">
  <script src="boxes.js"></script>
</head>
<body>
  <form id="data">
    <ol>
      <li>Pick a name for your Amazing Box: <br>
        <label for="name">Name: </label>
          <input type="text" id="name" size="20">
      </li>
      <li>Pick a color for your Amazing Box: <br>
        <select id="color">
          <option value="red">Red</option>
          <option value="orange">Orange</option>
          <option value="yellow">Yellow</option>
          <option value="green">Green</option>
          <option value="blue">Blue</option>
          <option value="indigo">Indigo</option>
          <option value="violet">Violet</option>
        </select>
      </li>
      <li>How many Amazing Boxes do you want to create?<br>
        <input type="radio" id="five" name="amount" value="5">
          <label for="five">5</label><br>
        <input type="radio" id="ten" name="amount" value="10">
          <label for="ten">10</label><br>
        <input type="radio" id="fifteen" name="amount" value="15">
          <label for="fifteen">15</label><br>
      </li>
    </ol>
    <p>
      <input type="button" id="generateButton" value="Generate!">
      <input type="button" id="clearButton" value="Clear">
    </p>
  </form>
  <div id="scene">
  </div>
</body>
</html>
```

Save this with an appropriate name (we suggest **boxes.html**) in your javascript homework folder. We provide links to the **boxes.css** and **boxes.js** files in the <head>; use these names for the files, or change them here to the names you use.

We have a form with three different input types: text input, select, and radio button input. You'll need to make sure that the text input and radio button inputs are valid (the color will always default to the first item in the list, so you don't technically need to check that one). Also, we have two buttons: one to generate the boxes, and one to clear the scene. And finally, we have a "scene" element at the very bottom; this is where all the boxes are going to go!

Look carefully over the HTML until you're familiar with each part.

The CSS for Amazing Boxes

Here is the CSS for the app:

CODE TO TYPE:

```
html, body {
    width: 100%;
    height: 90%;
    margin: 0px;
    padding: 0px;
    font-family: Helvetica, Arial, sans-serif;
}

input#generateButton {
    margin-left: 15px;
}

div#status {
    margin-left: 15px;
}

div#scene {
    position: relative;
    width: 100%;
    height: 80%;
    margin: 0px;
    padding: 0px;
}

div.box {
    position: absolute;
    width: 100px;
    height: 90px;
    padding-top: 10px;
    text-align: center;
    overflow: hidden;
}
```

Save it in your javascript homework folder as **boxes.css**, or again, use any name you want and change the link in the HTML file to the appropriate name. The most important two rules in the CSS are the **div#scene** rule and the **div.box** rule. Note that the "scene" is positioned relative, which means we can position the box s inside the "scene" using absolute positioning, and they will be positioned relative to the top left corner of the "scene" . That way, the boxes don't sit on top of the form!

When you add the elements that are going to represent the boxes, you'll need to add the class "box" to each element that represents a box. You already know how to do this.

The elements with the class "box" will be absolutely positioned, which means they can be positioned using the **top** and **left** properties. You can make these values anything between 0 and the height/width of the "scene" (which we'll show you how to do a moment).

Notice that the boxes are 100px by 100px (with 10px of padding on the height above the name of the box). The text is aligned to the center and any overflow text is hidden. That means when you add the name of the box as the content of the , it will appear in the center of the box.

Positioning an Absolutely Positioned Element

You will position each box (that is, each element with the class "box") within the "scene" by randomly

creating an x, y (left, top) position for the box. Here's how you can do that:

OBSERVE:

```
var sceneDiv = document.getElementById("scene");
var x = Math.floor(Math.random() * (sceneDiv.offsetWidth-101));
var y = Math.floor(Math.random() * (sceneDiv.offsetHeight-101));
```

We first get the **"scene"** from the DOM, and then use its computed width and height to generate a random number x, between 0 and the **width of the **, and a random number y, between 0 and the **height of the **. The computed width and height of the are figured out by the browser once the page is rendered, and you can get the values using the 's **offsetWidth** and **offsetHeight** properties. We don't want the boxes sitting halfway off the page at the side or bottom, so we reduce the possible starting point by the width + 1 of the box . Thus, if the computed width of the "scene" is 600px, the maximum x starting point for the left point of the box is 499 so the box will fit into the remaining part of the screen.

Once you have an x and y position, how do you update the style to set the position for the ? You can use the **style** property, which you already know how to do. However, there's one little tricky thing with the **top** and **left** properties you should know about. Here's what you'd do in CSS if you wanted to set these properties to 10, 10:

OBSERVE:

```
div.box {
    left: 10px;
    top: 10px;
}
```

So, in JavaScript, to set those same values (given x and y as computed above), you'd write:

OBSERVE:

```
div.style.left = x + "px";
div.style.top = y + "px";
```

We have to add the "px" after the values so the style properties for left and top match what you'd write in the CSS.

The Amazing Boxes Application Requirements

That's everything you need to build the app. Feel free to add any additional functions and variables you need. **Good luck!** Open the Objective to submit your files.