JavaScript
demystified

JavaScript demystified

JavaScript is a programming language (code) that lets you change the HTML (another kind of code) in a document on the fly, or "dynamically." This is useful for interactivity, animation, etc. When we use code to change code, we say we are "scripting." Whence the "Script" in the name "JavaScript."

So, say we have a web page with one image on it, and we want that image to change when clicked on, then change back when clicked on again. We will need to write a script for that. We'll design the script conceptually. Then, we'll see how it looks in JavaScript.

1. Express the program in plain English, as precisely as you can.

```
When I click on Apple.png, replace the image with Banana.png.
When I click on Banana.png, replace it with Apple.png.
```

This instruction in English says the same thing, but using only "if-then" statements:

```
When I click on the image,
      if the image is Apple.png, replace it with Banana.png,
      otherwise, replace it with Apple.png.
```

2. Try to express the same thing in "pseudo code." Pseudo code is just English in outline form that makes it easier to see the parts of the process and how they depend on each other. Try to use simple "if, then" statements. It's very loose, though, and just meant to help you conceptualize what you have to do. Use indents and alignment to help organize your thoughts. This "pseudo code" could be translated into ANY programming language.

Pseudo Code:

```
If image clicked, then

      if image is Apple.png, then
            replace with Banana.png;
      else replace with Apple.png
```

There are two parts to this. The first part is "If image clicked, then…" and the second part is another "if, then" statement, with an "else" appended. We can see that this is a set of "nested" instructions.

3. **Now, HTML includes simple mouse events (like clicking) that can trigger JavaScript functions. So the first part will be HTML. HTML needs to watch for a click "event" on the image, and trigger or "call" a JavaScript "function" in response. Let's name our JavaScript "function" something useful like "SwapImage."**

pseudo HTML:

```
If image is clicked, then call "SwapImage"
```

4. **Now we "define" the JavaScript function in pseudo code.**

```
SwapImage:

      if image is Apple.png, then
            replace with Banana.png;
      else, replace with Apple.png
```

5. Here's how we define the function in psuedo-code that looks a bit more like JavaScript. We have to say "function" first. A function name always includes parentheses at the end, so we add those. (You'll see what they're for next.) Curly Braces {} enclose the instructions that define the function.

```
function SwapImage()
      { do the whole swap thing }
```

We'll write nested instructions with nested brackets { {} }. Again, use indents and alignment to help organize your thoughts. An if-then-else statement looks like this in JavaScript...

...in general:

```
function SwapImage()
      { if this
            { do A }
            else
            { do B }
      }
```

...in our case:

```
function SwapImage()
      {  if image is Apple.png, then,
            { replace with Banana.png }
            else,
            { replace with Apple.png }
      }
```

JavaScript will have to get the name of the image from the HTML to know which one it is. So let's add a line:

```
function SwapImage()
            {  get the image;
                  if image is Apple.png, then,
                     { replace with Banana.png }
                     else,
                     { replace with Apple.png }
            }
```

6. We're ready for a sneak peak at the actual JavaScript + HTML version of this. The JavaScript will be enclosed in HTML <script></script> tags.

```
<img id="fruitpic" height="400px" src="Apple.png" onclick="SwapImage()" >

<script>

      function SwapImage()
            {
                  var fruit = document.getElementById('fruitpic');
                        if (fruit.src.match("Apple"))
                              { fruit.src = "Banana.png"; }
                        else
                              { fruit.src = "Apple.png"; }
            }

</script>
```

The script "gets" the image by looking for a named "id" in the HTML code. in this case the id is "fruitpic" and appears inside the <img> tag.

When someone clicks on the image, the script actually rewrites the HTML, inserting Banana.png in place of Apple.png. So the HTML now reads:

```
<img id="fruitpic" src="Banana.png" onclick="SwapImage()" >
```

So, the browser now displays the correct image. When the image is clicked again, the HTML is rewritten again.

7. It may help to see the actual JavaScript translated back into Pseudo-code:

```
function SwapImage()
      {
            declare a variable named “fruit,” make it = element in this document with id “fruitpic”;
                  if ( the “src” attribute of “fruit” matches “Apple” )
                        { change the src attribute of “fruit” to “Banana.png"; }
                  else
                        { change the src attribute of “fruit” to “Apple.png"; }
      }
```