jQuery: Ajax

1. Make a JSP page that returns a random number. Use $.ajax to make a pushbutton that, when pressed, gets the random number and pops it up in an alert box. Use an explicit onclick handler in the HTML.

2. Repeat the above problem, but avoid any explicit JavaScript in the HTML page (i.e., use the “unobtrusive JavaScript” approach that assigns to the click handler programmatically after the DOM is loaded).

3. Make a button that, when pressed, gets a random number from the server and inserts it into the page. Use “load”.

4. Make a form with three textfields (person1, person2, person3). Send the values to the server, and have the server send back a bulleted list (i.e., <ul> list) showing the names. Don’t use “serialize”.

5. Repeat the previous problem, but use “serialize”.

6. Make a form that collects a banking customer id and then, when a button is pressed, shows a list of the id, first name, last name, and balance. Give an error message for unknown ids. Use JSON on the server.

   You have done almost-identical tasks before, but use jQuery this time. If you never completed any of the previous Customer exercises, then just use a JSP page that returns a JSON object with three numbers in it:
   ```
   { num1: <%= Math.random() %>,
     num2: <%= Math.random() %>,
     num3: <%= Math.random() %>}
   ```
   Then, just do something simple with the result. The point is to get experience with $.ajax and the use of the dataType option.

7. Improve the previous example in two ways:

   • Show some sort of “working...” message while you are waiting for the response, and make the server response slow enough that you can see this in action.
   • Don’t hardcode the name of the result region in the JavaScript function, but instead pass it from the HTML page (not from the end user: from the HTML source code). You might think this is hard since the function you pass to onSuccess takes one argument only, but think about the way we have done response handlers in the past, and this should be very easy.