# Exercises: jQuery Ajax Basics

1. To start, make a new Dynamic Web project, go to my jquery-ajax-1 project, and copy jquery-test.html and the scripts folder from my WebContent folder to the WebContent folder of your new Dynamic Web Project. You can also copy my css folder if you want. Recall that jquery-test.html loads jquery.js but does not execute any jquery code. Deploy your app and bring up the jquery-test.html page. Open Firebug or the Chrome developer tools and verify that jQuery is loaded.

2. Here is the entire contents of a simple JSP page that outputs a random number:

   ```
   <%= Math.random() %>
   ```

   Note that the output of the JSP is a number *only*, not a Web page containing a number. Make a file in WebContent that ends in .jsp (e.g., random-num.jsp) that contains the single line above. Load the URL of the JSP page in a browser and verify you see a number. Reload the page and verify that the number changes.

3. Reload your HTML page that loads jquery.js. Bring up Firebug or the Chrome developer tools. Use $.ajax to call to the JSP page and pop up the result in an alert box. Do all the JavaScript interactively.

4. Change your HTML file to now load your own JavaScript file after it loads jquery.js. Make a single function that does the $.ajax process you just executed in the previous problem. Reload the HTML file, then call this function interactively.

5. Make a pushbutton that, when pressed, calls the function you defined in the previous problem. Use an explicit onclick handler in the HTML.

6. Repeat the above problem, but avoid any explicit JavaScript in the HTML for the button (i.e., use the "unobtrusive JavaScript" approach that assigns to the click handler programmatically after the DOM is loaded).

7. Make a button that, when pressed, gets a random number from the server and inserts it into the page. As before, try it first with interactive calls in Firebug or Chrome, then attach the behavior to the pushbutton.