JavaScript frameworks

Imagine the following scenario: A designer is starting a new project and her client is interested in adding an interactive photo gallery to the site. The designer also needs to create a form that requires JavaScript validation. Since the designer is new to JavaScript, she finds code she can use for the photo gallery and the form validation, and adds it to her page. The designer later gets another job similar to the first, and she decides to reuse the code from her first project, so she saves the JavaScript code into an external file.

The designer now has a reusable library of code she can add to future projects, but there are some potential problems with this approach:

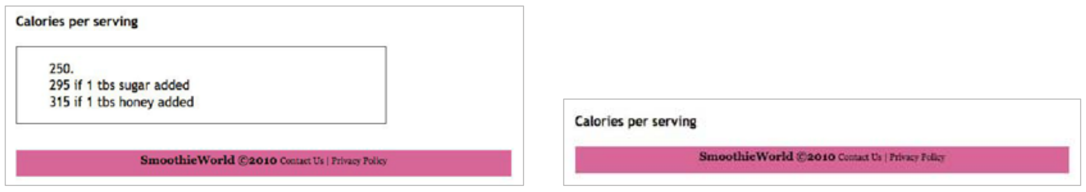
- The designer needs to organize, maintain, and update her library.
- The code the designer found could be poorly written.
- Poorly written JavaScript can slow down the performance of a website and/or cause security issues. Unless the designer is an expert at optimizing code, she may be unknowingly distributing bad code.

JavaScript frameworks are a better solution. There are several professionally written libraries available for use by designers. These libraries are large collections of functions built and tested by other designers and developers to form a common library. These collections of functions are available for immediate use, so if a designer needs to add an accordion menu (a menu that collapses and expands based on user events), he or she might readily find the code they need.

You will now use jQuery, one of the most popular and accessible JavaScript frameworks for designers. jQuery is useful for designers because it uses CSS syntax to search and access the page, thereby decreasing the amount of scripting language you need to learn.

Hiding an element with jQuery

In this exercise, you'll create an expandable container the user can toggle open and closed. The figures below show jQuery's animation features. The first image contains a box in its initial view; readers interested in the calorie content of the smoothie can click the *See calories* link to expand this section. The second image shows the expanded box after the user has clicked the button.



Calories per serving

250.
295 if 1 tbs sugar added
315 if 1 tbs honey added

SmoothieWorld ©2010 Contact Us | Privacy Policy

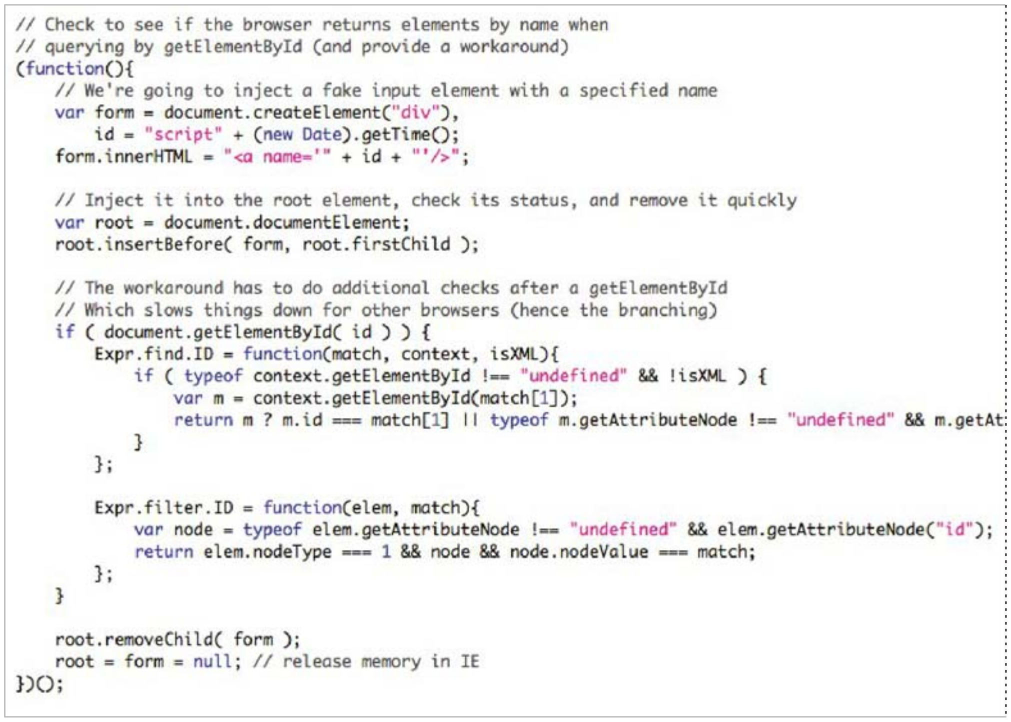
Calories per serving

SmoothieWorld ©2010 Contact Us | Privacy Policy

An example of the collapsible box you will create.

As you will see, jQuery lets you experiment with different methods of expanding the box and with the timing. The collapsible box will take two exercises to complete; in this first exercise, you will hide the section.

1 Perform this step to see the code of the jQuery framework. If you'd prefer to not see this, move on to step 2. Choose File > Open. In the dialog box that appears, navigate to your HTML5_08lessons and open the jquery.js file. Scroll to see the functions contained within the file. This file is well commented, so you can get a sense of what the functions do. When you are finished, close the file without saving it.



```
// Check to see if the browser returns elements by name when
// querying by getElementById (and provide a workaround)
(function(){
    // We're going to inject a fake input element with a specified name
    var form = document.createElement("div"),
        id = "script" + (new Date).getTime();
    form.innerHTML = "<a name='" + id + "'/>";

    // Inject it into the root element, check its status, and remove it quickly
    var root = document.documentElement;
    root.insertBefore( form, root.firstChild );

    // The workaround has to do additional checks after a getElementById
    // Which slows things down for other browsers (hence the branching)
    if ( document.getElementById( id ) ) {
        Expr.find.ID = function(match, context, isXML){
            if ( typeof context.getElementById !== "undefined" && !isXML ) {
                var m = context.getElementById(match[1]);
                return m ? m.id === match[1] || typeof m.getAttributeNode !== "undefined" && m.getAt
            }
        };

        Expr.filter.ID = function(elem, match){
            var node = typeof elem.getAttributeNode !== "undefined" && elem.getAttributeNode("id");
            return elem.nodeType === 1 && node && node.nodeValue === match;
        };
    }

    root.removeChild( form );
    root = form = null; // release memory in IE
})();
```

You can reference the functions in the jQuery document in your web page, but you rarely need to modify them.

2 Open the document jquerytoggle.html located in your HTML5_08lessons folder. Preview this page in your browser. The section of the page you will hide is the list below the heading *Calories per serving*. Close your browser and return to your document. Scroll to locate the HTML for this section; the list is wrapped in a `div` tag with the ID `CalorieBox`. This is the div you will hide.

3 In the jquerytoggle.html page, add the link to the jQuery JavaScript file, which is located in your HTML5_08lessons folder.

In the head section, immediately below the closing `</style>` tag, add the following code:

```
<script type="text/javascript" src="jquery.js"></script>
```

Choose File > Save. Your document can now access the functionality within the library. Note that this link to the jQuery library should go on every page that might reference code within it. Now you will add another script tag to add code that hides your Calories box.

4 Immediately below the `<script>` tag you just added, type the following code to add an empty `<script>` element:

```
<script type="text/javascript" src="jquery.js"></script>
<script>

</script>
```

You will now add a line of code that is included in almost every project that uses jQuery.

5 Add the following code into your empty `<script>` element:

```
<script>
  $(document).ready(function() {

  });
</script>
```

In this code, the `$` symbol is a reference to the jQuery object and `ready` is the event to respond to (in this case, the document being ready). In addition, you are defining a new function with `function`. This section of code is referred to as a ready function and prevents your code from searching the DOM until the document is fully loaded.

For example, in the following code, you will hide text on your page, so you want this code to be hidden when the page first loads.

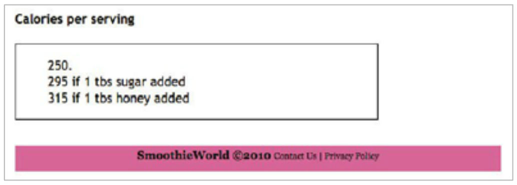
6 Scroll to locate the HTML code close to the bottom of the page that begins with the line `<div id="CalorieBox">`. This is the element on the page that you will hide; it contains a definition list that has the calorie values. jQuery allows objects in the DOM to be selected by several criteria. Since you want to select one specific element, you will search for that specific ID.

7 Scroll back up the page and add the following code immediately below your `document.ready` function:

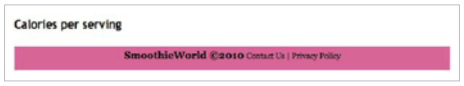
```
$(document).ready(function() {
  $('#CalorieBox').hide();
});
```

The hash tag (#) tells jQuery to search for an element with the ID `'CalorieBox'` (using the CSS selector syntax). Once found, jQuery will run the selected element's hide function, which is also a jQuery function.

8 Save your page, and then preview it in your browser. Your Calories section has disappeared from the page. Note that all the functionality for this effect is condensed in the line you added in the last step. This line works because the jQuery library is referenced in your HTML page.



The `CalorieBox` *before hiding it with jQuery*



The `CalorieBox` *after hiding it with jQuery*

This page lacks a trigger to cause the box to appear. You will now add this trigger by adding a link to the *Calories per serving* heading, as well as more jQuery code.

Adding an event to trigger the show effect

The effect you want to create is to expand the list currently hidden when the user clicks the *Calories per serving* heading. To do this, you will make the heading a link and give it an ID.

1 Locate the Calories per serving heading and add the following attributes:

```
<h4><a id="triggerCalorieBox" href="#">Calories per serving</a></h4>
```

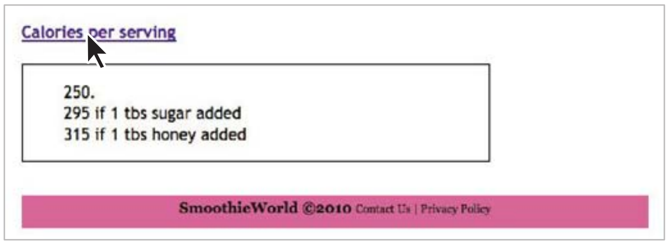
You are giving this heading an ID so you can target it with another line of jQuery. The href attribute is a dummy link that makes the heading a hyperlink, but is only there to serve as a trigger.

2 Scroll to your JavaScript code and add these four lines:

```
$(document).ready(function() {
  $('#CalorieBox').hide();
  $('a#triggerCalorieBox').click(function() {
    $('#CalorieBox').show();
    e.preventDefault()
  });
});
```

The first line identifies the hyperlinked ID you created in step 1 and attaches it to a click event. The second line is the instruction to show the `CalorieBox` ID. The third line is needed to override the default behavior of the hyperlink. (As previously noted, this hyperlink doesn't go to another page, so this line is necessary.) The fourth line is the closing bracket for the new function. (The opening bracket for this function is on the `.click(function()` line.)

3 Save your page and then preview it in your browser. Click the *Calories per serving* link; the box expands. The style for this box has been defined as 450 pixels wide with a black border on all sides.

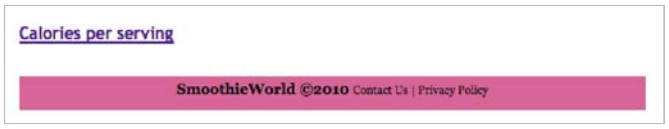


Clicking the link triggers the box to expand.

4 To enable the box to close again upon clicking, you need to add a line of code to hide the box after it has been expanded. The effect you want is for the user to toggle the box open and close by clicking the link. jQuery has a toggle effect you can use. You simply need to replace the show effect you have with the toggle. Replace the show effect with the following code:

```
$('a#triggerCalorieBox').click(function() {
  $('#CalorieBox').toggle();
  e.preventDefault();
```

5 Save your page and preview it in your browser. The CalorieBox is still hidden when the page loads. When you click it, it expands, and when you click again, it collapses. Close your browser and return to your text editor.



Using the toggle effect, the user can now open and close the box.

To make the show-and-hide effect more interesting, you will use the animation capabilities of jQuery.

6 In the lines of code you have already written, you can add control for the speed of the show-and-hide effect. Add the following code:

```
$('#CalorieBox').toggle('slow');
```

Save your page and then preview it in your browser. Clicking the link now results in a slow expansion of the box. If you want more precise control of the speed of the effect, jQuery allows you to control the speed using millisecond number values.

7 Return to your text editor and replace the `'slow'` value with a millisecond value (be sure to remove the single quotation marks, which are used for keywords such as `'slow'` or `'fast'`):

```
$('#CalorieBox').toggle(1200);
```

The 1200 milliseconds value is equivalent to 1.2 seconds. Save your page and then preview it in your browser. Clicking the link now results in a much slower expansion of the box. You'll now increase the speed of this effect.

8 Return to your text editor and replace the 1200 value with **500**, the equivalent of one-half second:

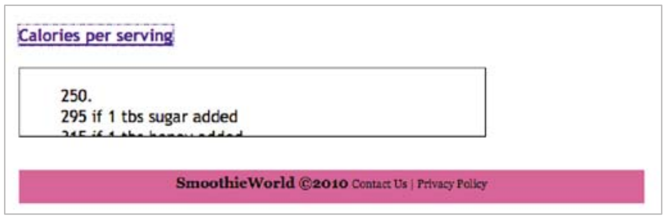
```
$('#CalorieBox').toggle(500);
```

You also have options to change the behavior of the box: in addition to `.show`, `.hide`, and `.toggle`, there are effects such as `.slideDown`, `.fadeIn`, and `.fadeOut`. You'll change your toggle effect to the slideToggle effect.

9 Add the following code:

```
$('#CalorieBox').slideToggle(500);
```

Save your page and preview it in your browser. When satisfied, close your browser and your file since you will be working with a new document in the next exercise.



The `slideToggle` *effect changes the behavior of the animation.*

As you can see, jQuery allows different options for your designs. The ability to show and hide animated elements is just one thing you can do with this library. The best way to learn more about jQuery is to go online to the source at *jQuery.com*, or to explore other online resources.

More advanced jQuery

jQuery and other similar JavaScript libraries are now used with increasing frequency on modern websites. User interface elements, such as drop-down or accordion menus, are two examples of these effects. You will also find jQuery used in slide shows, forms, multimedia, and much more.

As you explore jQuery, you will see that it supports plug-ins, which are sets of additional code that rely on some functionality in jQuery, and then build upon it. For example, in the previous exercise, the ability to control the speed of the box was limited. You could choose to start expanding the box slowly, and then speed up the expansion as it reached the end. In animation, this is referred to as easing, and several jQuery plugins have been created that give designers access to these effects.

Adding plug-ins involves adding another external JavaScript file to your site, and then linking to it. This adds new functions that you can then refer to in your HTML. For more information on plugins as well as documentation and examples, go to *http://plugins.jquery.com*.

Self study

1 Experiment with different effects for your jQuery calorie box. For example, replace the `.slideToggle` effect in the jquerytoggle.html page with the `.fadeIn` effect.

2 Experiment with the speed values in your code (currently set to 500 milliseconds) to see how they affect the behavior of the element.

3 Browse jquery.com, use the online interactive tutorials, and choose an example to integrate into your page.

Review

Questions

1 What is an event as it relates to JavaScript and HTML?

2 What is a JavaScript library, and what are the advantages of using one?

3 In this line of code, what does the number `500` stand for?

```
$('#CalorieBox').toggle(500);
```

Answers

1 An event on an HTML page often originates from a user interaction, such as clicking a button or loading a page. These events can then trigger specific JavaScript code that runs in the user's browser.

2 A JavaScript library, such as jQuery, is a collection of JavaScript code that lives inside an external JavaScript file. You can easily reference a library to add functionality, such as animated menus or user interface elements. Using a library is advantageous for designers because they can add relatively sophisticated behavior without writing complex JavaScript code.

3 Milliseconds.