Book 7: Object Model and Event Model

Book for Weeks 14-16: DHTML - Object Model and Event Model

Site: Insight2 @ CCSF
Course: CNIT133-META-Interactive Web Pages-Rubin-JavaScri-Spring 2014
Book: Book 7: Object Model and Event Model
Printed by: Thuong Ho
Date: Tuesday, April 29, 2014, 3:50 PM

Table of contents

1 Document Object Model

Textbook References

Flanagan book: Ch. 14 and Ch. 15

Gosselin book: p. 220-227

McDuffie book: Ch. 15

Deitel book: Ch. 12-13

The Document Object Model (DOM) is an application programming interface (API). The DOM enables any HTML tag to be seen as an object. A web page is seen as a single object, and the elements that compose it are seen as properties of that object. Each of the elements is also seen as an object, although a subordinate one, and the elements have their own properties, in turn. This process enables you to add scripting to any element, making the actions of the script specific to the actions of that element.

Here is more information about DOM.

2 Cross-browser & Cross-platform Considerations

All Web designers must be careful when catering to users with different browsers and/or platforms. One solution is to use a JavaScript browser sniffer.

You simply create a web page that ONLY contains a script that will test the user's browser and platform. So, if the user's browser is Internet Explorer with version 9 or less, and the user's platform is Windows then the script will open a window that may contain a web page with IE ActiveX controls. Otherwise the script will open a window that contains a web page without those features.

Here is an example coding of a browser sniffer.

```
<head>
<title>Browser and Platform Routing</title>


<script type="text/javascript">

function browserRouting()
{

bName = navigator.appName;
bVer = parseInt(navigator.appVersion);
pName = navigator.platform;

if (bName == "Microsoft Internet Explorer" && bVer <=9
&& pName == "Win32")

window.location='pathsind.htm';
else
window.location='index1.html';
}

</script>

</head>

<body onload="browserRouting()">

</body>
```

If your browser is IE and your platform is Windows then the script opens a window containing pathsind.htm. Otherwise, index1.html will be opened in a window.

Sometimes you must be careful when catering to users with different screen sizes. This is sometimes desirable when you have coded your HTML file using CSS absolute positioning. If you code the page for a screen size of 1280x800, a user with a higher screen size may see a fairly large area of space to the right of their screen. If you code the page for a screen size of 1440X900 or higher, a user with a lower screen size may see a horizontal scroll bar.

In this case you can use a JavaScript Screen sniffer. For example:

```
<script type="text/javascript">

if ((screen.width <= 1280) && (screen.height <= 800))

{

window.location = '1280-800-or-less.html';

}

else

{

 window.location = 'greater-than-1280-800.html';

}

</script>
```

3 One-Page FAQ using jQuery

The following is a tutorial that I have simplified from Chapter 5 of the book, JavaScipt and jQuery, the Missing Manual.

Tutorial: A One-Page FAQ

NOTE: This tutorial uses jQuery version 1.7.1

“Frequently Asked Questions” pages are a common sight on the web. They can help improve customer service by providing immediate answers 24/7. Unfortunately, most FAQ pages are either one very long page full of questions and complete answers, or a single page of questions that link to separate answer pages. Both solutions slow down the visitors’ quest for answers: in the first case, forcing a visitor to scroll down a long page for the question and answer she’s after, and in the second case, making the visitor wait for a new page to load.

In this tutorial, you’ll solve this problem by creating a JavaScript-driven FAQ page. All of the questions will be visible when the page loads, so it’s easy to locate a given question. The answers, however, are hidden until the question is clicked—then the desired answer fades smoothly into view.

Overview of the Task
The JavaScript for this task will need to accomplish several things:
• When a question is clicked, the corresponding answer will appear.
• When a question whose answer is visible is clicked, then the answer should disappear.

In addition, you’ll want to use JavaScript to hide all of the answers when the page loads. Why not just use CSS to hide the answers to begin with? For example, setting the CSS display property to none for the answers is another way to hide the answers. The problem with this technique is what happens to visitors who don’t have JavaScript turned on: They won’t see the answers when the page loads, nor will they be able to make them visible by clicking the questions. To make your pages viewable to both those with JavaScript enabled and those with JavaScript turned off, it’s best to use JavaScript to hide any page content.

Here is the FAQ page displayed:

http://fog.ccsf.cc.ca.us/~srubin/faq_page.html

Here is the jQuery code:

<script type="text/javascript" src="http://code.jquery.com/jquery-1.7.1.js"></script>

```
<script type="text/javascript">
$(document).ready(function() {
$('.answer').hide();
$('.main h2').toggle(
function() {
$(this).next('.answer').slideDown();
$(this).addClass('close');
},
function() {
$(this).next('.answer').fadeOut();
$(this).removeClass('close');
}
); // end toggle
}); // end ready
</script>
```

Here is the relevant HTML code:

```
<div class="main">
<h1>A One Page FAQ</h1>

<h2>I've heard that JavaScript is the long-lost fountain of youth. Is this true?</h2>
<div class="answer">
<p>Why, yes it is! Studies prove that learning JavaScript freshens the mind and extends life span by several hundred years. (Note: some scientists disagree with these claims.)</p>
</div>

<h2>Can JavaScript really solve all of my problems?</h2>
<div class="answer">
<p>Why, yes it can! It's the most versatile programming language ever created and is trained to provide financial management advice, life-saving CPR, and even to take care of household pets.</p>
</div>

<h2>Is there nothing JavaScript <em>can&amp;#8217;t</em> do?</h2>
<div class="answer">
<p>Why, no there isn&amp;#8217;t! It&amp;#8217;s even able to write its own public relations-oriented Frequently Asked Questions pages. Now that&amp;#8217;s one smart programming language!</p>
</div>
```

Again here is the jQuery code:

```
<script type="text/javascript" src="http://code.jquery.com/jquery-1.7.1.js"></script>
```

```
<script type="text/javascript">
$(document).ready(function() {
$('.answer').hide();
$('.main h2').toggle(
function() {
$(this).next('.answer').slideDown();
$(this).addClass('close');
},
function() {
$(this).next('.answer').fadeOut();
$(this).removeClass('close');
}
); // end toggle
}); // end ready
</script>
```

Explanations:

1. **<script type="text/javascript" src="http://code.jquery.com/jquery-1.7.1.js"></script>**

Get jquery library script

2. **<script type="text/javascript">**

Begin script for jquery code

3. **$(document).ready(function() {**

This is always needed as the first line in jquery code. The $(document).ready() function is a built-in jQuery
function that waits until the HTML for a page loads before it runs your script.

4. **$(‘.answer’).hide();**

The text of each answer is contained within a tag with the class of answer applied to it. This one line of code selects each and hides it. The answers should all be hidden.

The next step is determining which elements you need to add an event listener to. Since the answer appears when a visitor clicks the question, you must select every question in the FAQ. On this page, each question is a <h2> tag in the page’s main body.

5. **$('.main h2').toggle(**

The .main h2 a basic descendent selector used to target every <h2> tag inside an element with a class of main (so it doesn't affect any <h2> tags elsewhere on the page). Now it's time to add an event. The click event is a good candidate; however, you can better meet your requirements—that clicking the question either shows or hides the answer—using the jQuery toggle() function.
This function lets you switch between two different functions with each mouse click.

The toggle code marks the beginning of the toggle() function, which takes two anonymous functions as arguments. The first anonymous function runs on the first click, the second function runs on the next click. The basic structure of these functions is:

```
function() {
},
function() {
}
); // end toggle
}); // end ready
```

Be sure you don't leave out the comma at the end of the first function as the two functions here act like arguments passed to a function. In other words, the comma separates the two functions.

Now it's time to add the effect you're after: The first time the <h2> tag is clicked, the associated answer needs to appear. While each question is contained in a <h2> tag, the associated answer is in a tag immediately following the <h2> tag. In addition, the has a class of answer applied to it. So what you need is a way to find the tag following the clicked <h2>.

6. **$(this).next('.answer').slideDown();**

$(this) refers to the element currently responding to the event—in this case, a particular <h2> tag. jQuery provides several functions to make moving around a page's structure easier.

The .next() function finds the tag immediately following the current tag. In other words, it finds the tag following the <h2> tag. You can further refine this search by passing an additional selector to the .next() function—the code .next('.answer') finds the first tag following the <h2> that also has the class answer applied to it. Finally, .slideDown() gradually slides the answer into view.

In the next step, you'll complete the second half of the toggling effect—hiding the answer when the question is clicked a second time.

7. **$(this).addClass('close');**

This code simply adds a class named close to the <h2> tag when it's clicked the first time.

8. **$(this).next('.answer').fadeOut();**
$(this).removeClass('close');

This is the second half of the toggling effect — hiding the answer when the question is clicked a second time.The 2nd line removes that class when it's clicked a second time.

9. **Note - When viewing the page, notice the + and - signs.**

The plus sign is a common icon used to mean, "Hey, there's more here." To indicate that a visitor can click to hide the answer, replace the plus sign with a minus sign. You can do it easily by just adding and removing classes from the <h2> tags. The minus sign icon is defined within the style sheet as a background image. These signs are icons defined within the style sheet as background
images.

UPDATE:

Note that after jQuery version 1.7.1, the toggle method was deprecated (although it still works).
Michael has created the same example page using jQuery version 1.11.0
http://fog.ccsf.cc.ca.us/~srubin/faq_page2.html
The script:

```
<script type="text/javascript" src="http://code.jquery.com/jquery-1.11.0.min.js"></script>

<script type="text/javascript">
   $(document).ready(function () {
      $('.answer').hide();

      // Attach a click handler to the h2 elements.
      $(".main h2").click(function () {
         // Save the h2 element that was clicked
         // as well as the following answer element.
         var h2 = $(this);
         var answer = h2.next(".answer");
```

```
            if (answer.css("display") == "none") {
                // If the answer is currently not displayed,
                // display it with a sliding down motion,
                // and when done, add the "close" class to the h2 element.
                answer.slideDown(function () {
                    h2.addClass("close");
                });
            } else {
                // Otherwise, fade the answer out, and when done,
                // remove the "close" class from the h2 element.
                answer.fadeOut(function () {
                    h2.removeClass("close");
                });
            }
        });
    }); // end ready
</script>
```

4 Login Slider using jQuery

The following is a tutorial that I have simplified from Chapter 6 of the book, JavaScipt and jQuery, the Missing Manual.

Tutorial: Login Slider

NOTE: This tutorial uses jQuery version 1.7.1

In this tutorial, you'll get a little practice with using jQuery effects by creating a common user interface element: a panel that slides into and out of view with a click of the mouse.

The basic task is rather simple:

1. **Select the paragraph with the "Login" message on it.**
Remember that a lot of jQuery programming first begins with selecting an element on the page. In this case, the "Login" paragraph will receive clicks from a visitor.

2. **Attach a click event to that paragraph.**
JavaScript isn't interactive without events: The visitor needs to interact with the selection (the Login paragraph) to make something happen.

3. **Toggle the display of the form on and off.**
The previous two steps are just review (but necessary for so much of jQuery programming). This last step is where you'll use the effects you've learned about. You can make the form instantly appear (the show() function), slide into view (the slideDown() function), or fade into view (the fadeIn() function.)

Here is the login slider page displayed:

http://fog.ccsf.cc.ca.us/~srubin/loginjQ.html

NOTE: After jQuery version 1.7 the toggle method was deprecated.

Here is the jQuery code:

```
<script type="text/javascript" src="http://code.jquery.com/jquery-1.7.1.js"></script>
<script>
$(document).ready(function() {
$('#open').toggle(
function() {
$('#login form').slideDown(300);
$(this).addClass('close');
},
```

```
function() {
$('#login form').fadeOut(600);
$(this).removeClass('close');
}
); // end toggle
}); // end ready
</script>
```

Here is the relevant CSS code:

```
<style type="text/css">
#login {
width: 300px;
position: absolute;
left: 560px;
top: 6px;
z-index: 100;
}

#open {
margin: 0;
cursor: pointer;
background: rgb(255,255,255) url(open.png) no-repeat 8px 7px;
color: rgb(58,80,123);
padding: 5px 0 2px 30px;
}

#login .close {
background-image: url(close.png);
background-color: rgb(110,138,195);
}

#open:hover {
color: rgb(0,0,0);
background-color: rgb(110,138,195);
}

#login form {
padding: 10px 10px 10px 10px;
display: none;
background-color: rgb(255,255,255);
}
#login label {
display: inline-block;
width: 100px;
```

```
text-align:right;
margin: 0 15px 0 0;
color: rgb(58,80,123);
}
#login input {
font-size: 14px;
}
#login #button {
margin-left: 50px;
}
</style>
```

Here is the relevant HTML code:

```
<div id="login">
<p id="open">Login</p>
<form>
<p>
<label for="username">Username:</label>
<input type="text" name="username" id="username">
</p>
<p>
<label for="password">Password: </label>
<input type="password" name="password" id="password">
</p>
<p>
<input type="submit" name="button" id="button" value="Submit" >
</p>
</form>
</div>
```

Again, here is the jQuery code:

```
<script type="text/javascript" src="http://code.jquery.com/jquery-1.7.1.js"></script>
<script>
$(document).ready(function() {
$('#open').toggle(
function() {
$('#login form').slideDown(300);
$(this).addClass('close');
},
function() {
$('#login form').fadeOut(600);
$(this).removeClass('close');
```

```
}
); // end toggle
}); // end ready
</script>
```

Explanations:

1. **<script type="text/javascript" src="http://code.jquery.com/jquery-1.7.1.js"></script>**

Get jquery library script

2. **<script type="text/javascript">**

Begin script for jquery code

3. **$(document).ready(function() {**

This is always needed as the first line in jquery code. The $(document).ready() function is a built-in jQuery function that waits until the HTML for a page loads before it runs your script.

4. **$('#open').toggle(**

This code adds a click handler, so each time a visitor clicks on the paragraph, something happens. In this case, the form should appear when clicked once and then disappear when clicked again, appear on the next click, and so on. In other words, the form toggles between visible and invisible. jQuery offers three functions that will serve this purpose: toggle(), fadeToggle(), and slideToggle(). The difference is merely in how the effect looks.

5. **$('#login form').slideDown(300);**
$(this).addClass('close');

This code selects the login form, then slides it into view if it currently isn't visible, and then slides it back out of view if it is visible. Finally, you'll change the class of the paragraph, so that the "Login" can change appearance using a CSS class style.

When you're inside of an event handler, you can use $(this) to refer to the element that responds to the event. In this case, $(this) refers to the paragraph the visitor clicks on—the $('#open') in line 2 above. The toggleClass() function simply adds or removes a class from the element. Like the other toggle functions, toggleClass() adds the specified class if it's missing or removes the class if it's present. In this example, there's a class style—.close—in a style sheet on the page. (Look in the <head> of the file and you can see the style and what it does.)

6. **$('#login form').fadeOut(600);**
$(this).removeClass('close');

The above code will allow for fading out the effect.

Note that in this example you want two different effects - One for making the form appear —slide the form down into view, for example—and a different effect to make it disappear—fade out of view, for example. jQuery offers a special event—the toggle() event—for dealing with this kind of situation. Not to be confused with the toggle() effect—which makes an element appear and disappear—the toggle() event lets you run different code alternating between odd and even clicks. So on the first click, the form appears, and on the second click, it disappears.

UPDATE:

Note that after jQuery version 1.7.1, the toggle method was deprecated (although it still works).
Michael has created the same example page using jQuery version 1.11.0
http://fog.ccsf.cc.ca.us/~srubin/loginjQ2.html
The script:

```
<script type="text/javascript" src="http://code.jquery.com/jquery-1.11.0.min.js"></script>
  <script>
    $(document).ready(function () {
      // Attach a click handler to the element: <p id="open">
      $("#open").click(function () {
        // Save the <p id="open"> and <form> elements in local variables.
        var openElem = $(this);
        var loginForm = $("#login form");

        if (loginForm.css("display") == "none") {
          // If the form is currently not displayed,
          // slide it down into view, and when done add the "close" class to the
          // <p id="open"> element.
          loginForm.slideDown(300, function () {
            openElem.addClass("close");
          })
        } else {
          // Otherwise, fade the login form out, and when done,
          // remove the "close" class from the <p id="open"> element.
          loginForm.fadeOut(600, function () {
            openElem.removeClass("close");
          })
        }
      });
    }); // end ready
```

```
</script>
```

5 Web Tutorials and Examples - Object Model and Event Model

Document Object Model

- http://www.w3.org/TR/REC-DOM-Level-1/introduction.html

Navigator and screen object

- http://www.w3schools.com/jsref/obj_navigator.asp
- http://www.w3schools.com/jsref/obj_screen.asp

Broswer and screen size detection

- http://www.echoecho.com/jsbrowserdetection02.htm
- http://www.pageresource.com/jscript/jscreen.htm
- http://www.javascriptkit.com/howto/newtech3.shtml

getElementsByTagName Method

- http://www.java2s.com/Code/JavaScriptReference/Javascript-Methods/getElementsByTagName.htm

setTimeout and setInterval

- http://www.elated.com/articles/javascript-timers-with-settimeout-and-setinterval/
- http://www.w3schools.com/js/js_timing.asp
- http://fog.ccsf.cc.ca.us/~srubin/popup.htm

Navigation Bars

- http://net.tutsplus.com/tutorials/html-css-techniques/how-to-create-a-drop-down-nav-menu-with-html5-css3-and-jQuery/
- http://www.dynamicdrive.com/dynamicindex1/sm/index.htm
- http://www.javascriptkit.com/script/script2/verticalmenu.shtml

Event handling in the DOM

- http://wsabstract.com/dhtmltutors/domevent1.shtml

- http://www.brainjar.com/dhtml/events/

6 Coding Exercises - Object Model and Event Model

Exercises using Object Model and Event Model

Exercise 1:

Create a webpage that contains a script that utilizes the className property, select, radio and button elements to change the background, text and images on a webpage.

>>Here is the solution.

Exercise 2:

Create a password protection page using **jQuery version 1.7.1** that allows the user to click on a login button which will bring up the password prompt. The password is rosebud and is validated using javascript. If the user enters an incorrect password then display an error message. If the user enters a correct password then proceed to a form page.

>>Here is the solution.

Same problem except the password is validated using php and if correct then proceed to my homepage.

>>Here is the solution and the php script as a text file.

Exercise 3:

Create a page using jQuery containing a navigation bar of a Home link and three other general links. The first link should be Home. When you mouseover any of the three other links then display a drop-down menu that will contain more related links. See the source code for explanatory comments.

>>Here is the solution.

Exercise 4:

Create a webpage that contains a script that expands some text and an image upon loading.

>>Here is the solution.

Exercise 5:

Create a slide show using jQuery. For an explanation of the code see Chapter 7, starting on page 211.

>>Here is the solution.

Exercise 6:

Create a webpage containing a script that converts pound signs (#) in a form field to 'pound' when the field loses focus (user clicks outside of the field). Utilize the onblur event handler.

>>Here is a solution that uses a regular expression.

>>Here is another solution that uses charAt() and looping.

Exercise 7:

Create a webpage that contains a script that uses a button to center an image.

>>Here is the solution by Michael Ogi.