

```
1: <!--
2:
3: ajax1.html
4:
5: Gets stock quote from quotel.php via Ajax, displaying result with alert().
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html PUBLIC
13:   "-//W3C//DTD XHTML 1.0 Transitional//EN"
14:   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
15:
16: <html xmlns="http://www.w3.org/1999/xhtml">
17:   <head>
18:     <script type="text/javascript">
19:       // 
20:
21:         // an XMLHttpRequest
22:         var xhr = null;
23:
24:         /*
25:          * void
26:          * quote()
27:          *
28:          * Gets a quote.
29:          */
30:         function quote()
31:         {
32:             // instantiate XMLHttpRequest object
33:             try
34:             {
35:                 xhr = new XMLHttpRequest();
36:             }
37:             catch (e)
38:             {
39:                 xhr = new ActiveXObject("Microsoft.XMLHTTP");
40:             }
41:
42:             // handle old browsers
43:             if (xhr == null)
44:             {
45:                 alert("Ajax not supported by your browser!");
46:                 return;
47:             }
</pre></div><div data-bbox="117 524 227 551" data-label="Page-Header">ajax1.html<br/>lectures/9/src/ajax/</div><div data-bbox="849 524 883 539" data-label="Page-Header">2/2</div><div data-bbox="137 556 690 923" data-label="Text"><pre>48:
49:         // construct URL
50:         var url = "quotel.php?symbol=" + document.getElementById("symbol").value;
51:
52:         // get quote
53:         xhr.onreadystatechange = handler;
54:         xhr.open("GET", url, true);
55:         xhr.send(null);
56:     }
57:
58:
59:     /*
60:      * void
61:      * handler()
62:      *
63:      * Handles the Ajax response.
64:      */
65:     function handler()
66:     {
67:         // only handle loaded requests
68:         if (xhr.readyState == 4)
69:         {
70:             // display response if possible
71:             if (xhr.status == 200)
72:                 alert(xhr.responseText);
73:             else
74:                 alert("Error with Ajax call!");
75:         }
76:     }
77:
78: // ]]&gt;
79: &lt;/script&gt;
80: &lt;title&gt;&lt;/title&gt;
81: &lt;/head&gt;
82: &lt;body&gt;
83:   &lt;form action="" onsubmit="quote(); return false;"&gt;
84:     Symbol: &lt;input id="symbol" type="text" /&gt;
85:     &lt;br /&gt;&lt;br /&gt;
86:     &lt;input type="submit" value="Get Quote" /&gt;
87:   &lt;/form&gt;
88: &lt;/body&gt;
89: &lt;/html&gt;
</pre></div>
```

```
1: <!--
2:
3: ajax2.html
4:
5: Gets stock quote from quotel.php via Ajax, embedding result in page itself.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html PUBLIC
13:   "-//W3C//DTD XHTML 1.0 Transitional//EN"
14:   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
15:
16: <html xmlns="http://www.w3.org/1999/xhtml">
17:   <head>
18:     <script type="text/javascript">
19:       // 
20:
21:         // an XMLHttpRequest
22:         var xhr = null;
23:
24:         /*
25:          * void
26:          * quote()
27:          *
28:          * Gets a quote.
29:          */
30:         function quote()
31:         {
32:           // instantiate XMLHttpRequest object
33:           try
34:           {
35:             xhr = new XMLHttpRequest();
36:           }
37:           catch (e)
38:           {
39:             xhr = new ActiveXObject("Microsoft.XMLHTTP");
40:           }
41:
42:           // handle old browsers
43:           if (xhr == null)
44:           {
45:             alert("Ajax not supported by your browser!");
46:             return;
47:           }
</pre></div><div data-bbox="117 524 227 551" data-label="Page-Header">ajax2.html<br/>lectures/9/src/ajax/</div><div data-bbox="850 524 883 539" data-label="Page-Header">2/2</div><div data-bbox="137 556 690 941" data-label="Text"><pre>48:
49:         // construct URL
50:         var url = "quotel.php?symbol=" + document.getElementById("symbol").value;
51:
52:         // get quote
53:         xhr.onreadystatechange = handler;
54:         xhr.open("GET", url, true);
55:         xhr.send(null);
56:     }
57:
58:
59:     /*
60:      * void
61:      * handler()
62:      *
63:      * Handles the Ajax response.
64:      */
65:     function handler()
66:     {
67:       // only handle loaded requests
68:       if (xhr.readyState == 4)
69:       {
70:         // embed response in page if possible
71:         if (xhr.status == 200)
72:           document.getElementById("price").innerHTML = xhr.responseText;
73:         else
74:           alert("Error with Ajax call!");
75:       }
76:     }
77:
78: // ]]&gt;
79: &lt;/script&gt;
80: &lt;title&gt;&lt;/title&gt;
81: &lt;/head&gt;
82: &lt;body&gt;
83:   &lt;form action="" onsubmit="quote(); return false;"&gt;
84:     Symbol: &lt;input id="symbol" type="text" /&gt;
85:     &lt;br /&gt;
86:     Price: &lt;span id="price"&gt;&lt;b&gt;to be determined&lt;/b&gt;&lt;/span&gt;
87:     &lt;br /&gt;&lt;br /&gt;
88:     &lt;input type="submit" value="Get Quote" /&gt;
89:   &lt;/form&gt;
90: &lt;/body&gt;
91: &lt;/html&gt;
</pre></div>
```

```
1: <!--
2:
3: ajax3.html
4:
5: Gets stock quote (plus day's low and high) from quote2.php via Ajax,
6: embedding result in page itself after indicating progress with an
7: animated GIF.
8:
9: Computer Science 50
10: David J. Malan
11:
12: -->
13:
14: <!DOCTYPE html PUBLIC
15:   "-//W3C//DTD XHTML 1.0 Transitional//EN"
16:   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
17:
18: <html xmlns="http://www.w3.org/1999/xhtml">
19:   <head>
20:     <script type="text/javascript">
21:       // 
22:
23:         // an XMLHttpRequest
24:         var xhr = null;
25:
26:         /*
27:          * void
28:          * quote()
29:          *
30:          * Gets a quote.
31:          */
32:         function quote()
33:         {
34:           // instantiate XMLHttpRequest object
35:           try
36:           {
37:             xhr = new XMLHttpRequest();
38:           }
39:           catch (e)
40:           {
41:             xhr = new ActiveXObject("Microsoft.XMLHTTP");
42:           }
43:
44:           // handle old browsers
45:           if (xhr == null)
46:           {
47:             alert("Ajax not supported by your browser!");</pre></div><div data-bbox="117 524 227 551" data-label="Page-Header">ajax3.html<br/>lectures/9/src/ajax/</div><div data-bbox="849 524 883 539" data-label="Page-Header">2/3</div><div data-bbox="137 556 690 968" data-label="Text"><pre>48:             return;
49:           }
50:
51:           // construct URL
52:           var url = "quote2.php?symbol=" + document.getElementById("symbol").value;
53:
54:           // show progress
55:           document.getElementById("progress").style.display = "block";
56:
57:           // get quote
58:           xhr.onreadystatechange = handler;
59:           xhr.open("GET", url, true);
60:           xhr.send(null);
61:         }
62:
63:
64:         /*
65:          * void
66:          * handler()
67:          *
68:          * Handles the Ajax response.
69:          */
70:         function handler()
71:         {
72:           // only handle requests in "loaded" state
73:           if (xhr.readyState == 4)
74:           {
75:             // hide progress
76:             document.getElementById("progress").style.display = "none";
77:
78:             // embed response in page if possible
79:             if (xhr.status == 200)
80:               document.getElementById("quote").innerHTML = xhr.responseText;
81:             else
82:               alert("Error with Ajax call!");
83:           }
84:         }
85:
86:       // ]]&gt;
87:     &lt;/script&gt;
88:   &lt;title&gt;&lt;/title&gt;
89: &lt;/head&gt;
90: &lt;body&gt;
91:   &lt;form action="" onsubmit="quote(); return false;"&gt;
92:     Symbol: &lt;input id="symbol" type="text" /&gt;
93:     &lt;br /&gt;&lt;br /&gt;
94:     &lt;div id="progress" style="display: none;"&gt;</pre></div>
```

```
95:         
96:         <br /><br />
97:     </div>
98:     <div id="quote"></div>
99:     <br /><br />
100:     <input type="submit" value="Get Quote" />
101: </form>
102: </body>
103: </html>
```

```
1: <!--
2:
3: ajax4.html
4:
5: Gets stock quote from quote1.php via Ajax, displaying result with alert().
6: Implements handler as an anonymous function.
7:
8: Computer Science 50
9: David J. Malan
10:
11: -->
12:
13: <!DOCTYPE html PUBLIC
14:     "-//W3C//DTD XHTML 1.0 Transitional//EN"
15:     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
16:
17: <html xmlns="http://www.w3.org/1999/xhtml">
18:   <head>
19:     <script type="text/javascript">
20:       // 
21:
22:         // an XMLHttpRequest
23:         var xhr = null;
24:
25:         /*
26:          * void
27:          * quote()
28:          *
29:          * Gets a quote.
30:          */
31:         function quote()
32:         {
33:             // instantiate XMLHttpRequest object
34:             try
35:             {
36:                 xhr = new XMLHttpRequest();
37:             }
38:             catch (e)
39:             {
40:                 xhr = new ActiveXObject("Microsoft.XMLHTTP");
41:             }
42:
43:             // handle old browsers
44:             if (xhr == null)
45:             {
46:                 alert("Ajax not supported by your browser!");
47:                 return;</pre></div>
```

```
48:         }
49:
50:         // construct URL
51:         var url = "quote1.php?symbol=" + document.getElementById("symbol").value;
52:
53:         // get quote
54:         xhr.onreadystatechange = function () {
55:             // only handle loaded requests
56:             if (xhr.readyState == 4)
57:             {
58:                 // display response if possible
59:                 if (xhr.status == 200)
60:                     alert(xhr.responseText);
61:                 else
62:                     alert("Error with Ajax call!");
63:             }
64:         };
65:         xhr.open("GET", url, true);
66:         xhr.send(null);
67:     }
68:
69:     // ]]>
70: </script>
71: </title></title>
72: </head>
73: <body>
74:     <form action="" onsubmit="quote(); return false;">
75:         Symbol: <input id="symbol" type="text" />
76:         <br /><br />
77:         <input type="submit" value="Get Quote" />
78:     </form>
79: </body>
80: </html>
```

```
1: <!--
2:
3: ajax5.html
4:
5: Gets stock quote (plus day's low and high) from quote3.php via Ajax,
6: embedding (JSON) result in page itself.
7:
8: Computer Science 50
9: David J. Malan
10:
11: -->
12:
13: <!DOCTYPE html PUBLIC
14:     "-//W3C//DTD XHTML 1.0 Transitional//EN"
15:     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
16:
17: <html xmlns="http://www.w3.org/1999/xhtml">
18:     <head>
19:         <script type="text/javascript">
20:             // 
21:
22:                 // an XMLHttpRequest
23:                 var xhr = null;
24:
25:                 /*
26:                 * void
27:                 * quote()
28:                 *
29:                 * Gets a quote.
30:                 */
31:                 function quote()
32:                 {
33:                     // instantiate XMLHttpRequest object
34:                     try
35:                     {
36:                         xhr = new XMLHttpRequest();
37:                     }
38:                     catch (e)
39:                     {
40:                         xhr = new ActiveXObject("Microsoft.XMLHTTP");
41:                     }
42:
43:                     // handle old browsers
44:                     if (xhr == null)
45:                     {
46:                         alert("Ajax not supported by your browser!");
47:                         return;</pre></div>
```

```
48:         }
49:
50:         // construct URL
51:         var url = "quote3.php?symbol=" + document.getElementById("symbol").value;
52:
53:         // get quote
54:         xhr.onreadystatechange = function() {
55:
56:             // only handle requests in "loaded" state
57:             if (xhr.readyState == 4)
58:             {
59:                 // embed response in page if possible
60:                 if (xhr.status == 200)
61:                 {
62:                     var quote = eval("(" + xhr.responseText + ")");
63:                     document.getElementById("price").innerHTML = quote.price;
64:                     document.getElementById("high").innerHTML = quote.high;
65:                     document.getElementById("low").innerHTML = quote.low;
66:                 }
67:                 else
68:                     alert("Error with Ajax call!");
69:             }
70:
71:         };
72:         xhr.open("GET", url, true);
73:         xhr.send(null);
74:     }
75:
76:
77:     /*
78:     * void
79:     * handler()
80:     *
81:     * Handles the Ajax response.
82:     */
83:     function handler()
84:     {
85:     }
86:
87:     // ]]>
88: </script>
89: <title></title>
90: </head>
91: <body>
92: <form action="" onsubmit="quote(); return false;">
93:     Symbol: <input id="symbol" type="text" />
94:     <br /><br />
```

```
95:     Price: <span id="price"></span>
96:     <br />
97:     High: <span id="high"></span>
98:     <br />
99:     Low: <span id="low"></span>
100:     <br />
101:     <br /><br />
102:     <input type="submit" value="Get Quote" />
103: </form>
104: </body>
105: </html>
```

```
1: <?php
2:
3:  /**
4:   * quote1.php
5:   *
6:   * Outputs price of given symbol as text/html.
7:   *
8:   * Computer Science 50
9:   * David J. Malan
10:  */
11:
12:  // get quote
13:  $handle = @fopen("http://download.finance.yahoo.com/d/quotes.csv?s={$_GET['symbol']}&f=e1l1", "r");
14:  if ($handle !== FALSE)
15:  {
16:      $data = fgetcsv($handle);
17:      if ($data !== FALSE && $data[0] == "N/A")
18:          print($data[1]);
19:      fclose($handle);
20:  }
21: ?>
```

```
1: <?php
2:
3:  /**
4:   * quote2.php
5:   *
6:   * Outputs price, low, and high of given symbol as text/html, after
7:   * inserting an artificial delay.
8:   *
9:   * Computer Science 50
10:  * David J. Malan
11:  */
12:
13:  // pretend server is slow
14:  sleep(5);
15:
16:  // try to get quote
17:  $handle = @fopen("http://download.finance.yahoo.com/d/quotes.csv?s={$_GET['symbol']}&f=e1l1hg", "r");
18:  if ($handle !== FALSE)
19:  {
20:      $data = fgetcsv($handle);
21:      if ($data !== FALSE && $data[0] == "N/A")
22:      {
23:          print("Price: {$data[1]}");
24:          print("<br />");
25:          print("High: {$data[2]}");
26:          print("<br />");
27:          print("Low: {$data[3]}");
28:      }
29:      fclose($handle);
30:  }
31:
32: ?>
```

```
1: <?php
2:
3:     /**
4:     * quote3.php
5:     *
6:     * Outputs price, low, and high of given symbol as JSON.
7:     *
8:     * Computer Science 50
9:     * David J. Malan
10:    */
11:
12:    // try to get quote
13:    $quote = array();
14:    $handle = @fopen("http://download.finance.yahoo.com/d/quotes.csv?s={$_GET['symbol']}&f=e1l1hg", "r");
15:    if ($handle !== FALSE)
16:    {
17:        $data = fgetcsv($handle);
18:        if ($data !== FALSE && $data[0] == "N/A")
19:        {
20:            $quote["price"] = $data[1];
21:            $quote["high"] = $data[2];
22:            $quote["low"] = $data[3];
23:        }
24:        fclose($handle);
25:    }
26:    header("Content-type: text/javascript");
27:    print(json_encode($quote));
28: ?>
```

```
1: <?
2:
3:     /**
4:     * dump.php
5:     *
6:     * Dumps HTTP requests (albeit as invalid XHTML).
7:     *
8:     * Computer Science 50
9:     * David J. Malan
10:    */
11:
12: ?>
13:
14: <pre>
15: <?php print_r($_GET); ?>
16: </pre>
```



```
1: <!--
2:
3: form1.html
4:
5: A form without client-side validation.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html PUBLIC
13:   "-//W3C//DTD XHTML 1.0 Transitional//EN"
14:   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
15:
16: <html xmlns="http://www.w3.org/1999/xhtml">
17:   <head>
18:     <title></title>
19:   </head>
20:   <body>
21:     <form action="dump.php" method="get">
22:       Email: <input name="email" type="text" />
23:       <br />
24:       Password: <input name="password1" type="password" />
25:       <br />
26:       Password (again): <input name="password2" type="password" />
27:       <br />
28:       I agree to the terms and conditions: <input name="agreement" type="checkbox" />
29:       <br /><br />
30:       <input type="submit" value="Submit" />
31:     </form>
32:   </body>
33: </html>
```

```
1: <!--
2:
3: form2.html
4:
5: A form with client-side validation.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html PUBLIC
13:   "-//W3C//DTD XHTML 1.0 Transitional//EN"
14:   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
15:
16: <html xmlns="http://www.w3.org/1999/xhtml">
17:   <head>
18:     <script type="text/javascript">
19:       // 
20:
21:         function validate()
22:         {
23:           if (document.forms.registration.email.value == "")
24:             {
25:               alert("You must provide an email address.");
26:               return false;
27:             }
28:           else if (document.forms.registration.password1.value == "")
29:             {
30:               alert("You must provide a password.");
31:               return false;
32:             }
33:           else if (document.forms.registration.password1.value != document.forms.registration.password2.value)
34:             {
35:               alert("You must provide the same password twice.");
36:               return false;
37:             }
38:           else if (!document.forms.registration.agreement.checked)
39:             {
40:               alert("You must agree to our terms and conditions.");
41:               return false;
42:             }
43:           return true;
44:         }
45:
46:       // ]]&gt;
47:     &lt;/script&gt;</pre></div>
```

```
48:     <title></title>
49: </head>
50: <body>
51:   <form action="dump.php" method="get" name="registration"
52:     onsubmit="return validate();">
53:     Email: <input name="email" type="text" />
54:     <br />
55:     Password: <input name="password1" type="password" />
56:     <br />
57:     Password (again): <input name="password2" type="password" />
58:     <br />
59:     I agree to the terms and conditions: <input name="agreement" type="checkbox" />
60:     <br /><br />
61:     <input type="submit" value="Submit" />
62:   </form>
63: </body>
64: </html>
```

```
1: <!--
2:
3: form3.html
4:
5: A form with client-side validation demonstrating "this" keyword.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html PUBLIC
13:   "-//W3C//DTD XHTML 1.0 Transitional//EN"
14:   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
15:
16: <html xmlns="http://www.w3.org/1999/xhtml">
17:   <head>
18:     <script type="text/javascript">
19:       // 
20:
21:         function validate(f)
22:         {
23:           if (f.email.value == "")
24:           {
25:             alert("You must provide an email address.");
26:             return false;
27:           }
28:           else if (f.password1.value == "")
29:           {
30:             alert("You must provide a password.");
31:             return false;
32:           }
33:           else if (f.password1.value != f.password2.value)
34:           {
35:             alert("You must provide the same password twice.");
36:             return false;
37:           }
38:           else if (!f.agreement.checked)
39:           {
40:             alert("You must agree to our terms and conditions.");
41:             return false;
42:           }
43:           return true;
44:         }
45:
46:       // ]]&gt;
47:     &lt;/script&gt;</pre></div>
```

```
48:     <title></title>
49: </head>
50: <body>
51:   <form action="dump.php" method="get" name="registration" onsubmit="return validate(this);">
52:     Email: <input name="email" type="text" />
53:     <br />
54:     Password: <input name="password1" type="password" />
55:     <br />
56:     Password (again): <input name="password2" type="password" />
57:     <br />
58:     I agree to the terms and conditions: <input name="agreement" type="checkbox" />
59:     <br /><br />
60:     <input type="submit" value="Submit" />
61:   </form>
62: </body>
63: </html>
```

```
1: <!--
2:
3: form4.html
4:
5: A form with client-side validation demonstrating disabled property.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html PUBLIC
13:   "-//W3C//DTD XHTML 1.0 Transitional//EN"
14:   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
15:
16: <html xmlns="http://www.w3.org/1999/xhtml">
17:   <head>
18:     <script type="text/javascript">
19:       // 
20:
21:         function toggle()
22:         {
23:           if (document.forms.registration.button.disabled)
24:             document.forms.registration.button.disabled = false;
25:           else
26:             document.forms.registration.button.disabled = true;
27:         }
28:
29:         function validate()
30:         {
31:           if (document.forms.registration.email.value == "")
32:           {
33:             alert("You must provide an email address.");
34:             return false;
35:           }
36:           else if (document.forms.registration.password1.value == "")
37:           {
38:             alert("You must provide a password.");
39:             return false;
40:           }
41:           else if (document.forms.registration.password1.value != document.forms.registration.password2.value)
42:           {
43:             alert("You must provide the same password twice.");
44:             return false;
45:           }
46:           else if (!document.forms.registration.agreement.checked)
47:           {</pre></div>
```

```
48:         alert("You must agree to our terms and conditions.");
49:         return false;
50:     }
51:     return true;
52: }
53:
54: // ]]>
55: </script>
56: <title></title>
57: </head>
58: <body>
59:   <form action="dump.php" method="get" name="registration" onsubmit="return validate();">
60:     Email: <input name="email" type="text" />
61:     <br />
62:     Password: <input name="password1" type="password" />
63:     <br />
64:     Password (again): <input name="password2" type="password" />
65:     <br />
66:     I agree to the terms and conditions: <input name="agreement" onclick="toggle();" type="checkbox" />
67:     <br /><br />
68:     <input disabled="disabled" name="button" type="submit" value="Submit" />
69:   </form>
70: </body>
71: </html>
```

```
1: <!--
2:
3: form5.html
4:
5: A form with client-side validation demonstrating regular expressions.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html PUBLIC
13:   "-//W3C//DTD XHTML 1.0 Transitional//EN"
14:   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
15:
16: <html xmlns="http://www.w3.org/1999/xhtml">
17:   <head>
18:     <script type="text/javascript">
19:       // 
20:
21:         function validate()
22:         {
23:             if (!document.forms.registration.email.value.match(/.+@.+\.edu$/))
24:             {
25:                 alert("You must provide a .edu email address.");
26:                 return false;
27:             }
28:             else if (document.forms.registration.password1.value == "")
29:             {
30:                 alert("You must provide a password.");
31:                 return false;
32:             }
33:             else if (document.forms.registration.password1.value != document.forms.registration.password2.value)
34:             {
35:                 alert("You must provide the same password twice.");
36:                 return false;
37:             }
38:             else if (!document.forms.registration.agreement.checked)
39:             {
40:                 alert("You must agree to our terms and conditions.");
41:                 return false;
42:             }
43:             return true;
44:         }
45:
46:       // ]]&gt;
47:     &lt;/script&gt;</pre></div>
```

```
48:     <title></title>
49: </head>
50: <body>
51:     <form action="dump.php" method="get" name="registration" onsubmit="return validate();">
52:         Email: <input name="email" type="text" />
53:         <br />
54:         Password: <input name="password1" type="password" />
55:         <br />
56:         Password (again): <input name="password2" type="password" />
57:         <br />
58:         I agree to the terms and conditions: <input name="agreement" type="checkbox" />
59:         <br /><br />
60:         <input type="submit" value="Submit" />
61:     </form>
62: </body>
63: </html>
```

```
1: <?php
2:
3:     /*****
4:     * dictionary.php
5:     *
6:     * Computer Science 50
7:     * David J. Malan
8:     *
9:     * Implements a dictionary.
10:    *****/
11:
12:
13:     // size of dictionary
14:     $size = 0;
15:
16:     // dictionary
17:     $dictionary = array();
18:
19:
20:     /*
21:     * bool
22:     * check($word)
23:     *
24:     * Returns true if word is in dictionary else false.
25:     */
26:
27:     function check($word)
28:     {
29:         global $dictionary;
30:         if ($dictionary[strtolower($word)])
31:             return true;
32:         else
33:             return false;
34:     }
35:
36:
37:     /*
38:     * bool
39:     * load($dict)
40:     *
41:     * Loads dict into memory. Returns true if successful else false.
42:     */
43:
44:     function load($dict)
45:     {
46:         global $dictionary, $size;
47:         if (!file_exists($dict) && is_readable($dict))
```

```
48:         return false;
49:         foreach (file($dict) as $word)
50:             $dictionary[chop($word)] = true;
51:         return true;
52:     }
53:
54:
55:     /*
56:     * int
57:     * size()
58:     *
59:     * Returns number of words in dictionary if loaded else 0 if not yet loaded.
60:     */
61:
62:     function size()
63:     {
64:         global $size;
65:         return $size;
66:     }
67:
68:
69:     /*
70:     * int
71:     * unload()
72:     *
73:     * Unloads dictionary from memory. Returns true if successful else false.
74:     */
75:
76:     function unload()
77:     {
78:         return true;
79:     }
80: ?>
```

```
1: #!/usr/local/bin/php
2: <?php
3:
4:     /*****
5:     * speller.php
6:     *
7:     * Computer Science 50
8:     * David J. Malan
9:     *
10:    * Implements a spell-checker.
11:    *****/
12:
13:     require("dictionary.php");
14:
15:     // suppress notices and warnings
16:     error_reporting(E_ALL ^ E_NOTICE ^ E_WARNING);
17:
18:
19:     // maximum length for a word
20:     // (e.g., pneumonoultramicroscopicsilicovolcanoconiosis)
21:     define("LENGTH", 45);
22:
23:     // default dictionary
24:     define("WORDS", "/home/cs50/pub/share/pset6/dict/words");
25:
26:     // check for correct number of args
27:     if ($argc != 2 && $argc != 3)
28:     {
29:         print("Usage: speller.php [dict] file\n");
30:         return 1;
31:     }
32:
33:     // benchmarks
34:     $ti_load = 0.; $ti_check = 0.; $ti_size = 0.; $ti_unload = 0.;
35:
36:     // determine dictionary to use
37:     $dict = ($argc == 3) ? $argv[1] : WORDS;
38:
39:     // load dictionary
40:     $before = microtime(TRUE);
41:     $loaded = load($dict);
42:     $after = microtime(TRUE);
43:
44:     // abort if dictionary not loaded
45:     if (!$loaded)
46:     {
47:         print("Could not load $dict.\n");
```

```

48:         return 2;
49:     }
50:
51:     // calculate time to load dictionary
52:     $ti_load = $after - $before;
53:
54:     // try to open file
55:     $file = ($argc == 3) ? $argv[2] : $argv[1];
56:     $fp = fopen($file, "r");
57:     if ($fp === FALSE)
58:     {
59:         print("Could not open $file.\n");
60:         return 3;
61:     }
62:
63:     // prepare to report misspellings
64:     printf("\nMISPELLED WORDS\n\n");
65:
66:     // prepare to spell-check
67:     $word = "";
68:     $index = 0; $misspellings = 0; $words = 0;
69:
70:     // spell-check each word in file
71:     for ($c = fgetc($fp); $c != FALSE; $c = fgetc($fp))
72:     {
73:         // allow alphabetical characters and apostrophes (for possessives)
74:         if (preg_match("/[a-zA-Z]/", $c) || ($c == "'" && $index > 0))
75:         {
76:             // append character to word
77:             $word .= $c;
78:             $index++;
79:
80:             // ignore alphabetical strings too long to be words
81:             if ($index >= LENGTH)
82:             {
83:                 // consume remainder of alphabetical string
84:                 while (($c = fgetc($fp)) != FALSE && preg_match("/[a-zA-Z]/", $c));
85:
86:                 // prepare for new word
87:                 $index = 0; $word = "";
88:             }
89:         }
90:
91:         // ignore words with numbers (like MS Word)
92:         else if (ctype_digit($c))
93:         {
94:             // consume remainder of alphabetical string

```

```

95:         while (($c = fgetc($fp)) != FALSE && preg_match("/[a-zA-z0-9]", $c));
96:
97:         // prepare for new word
98:         $index = 0; $word = "";
99:     }
100:
101:     // we must have found a whole word
102:     else if ($index > 0)
103:     {
104:         // update counter
105:         $words++;
106:
107:         // check word's spelling
108:         $before = microtime(TRUE);
109:         $misspelled = !check($word);
110:         $after = microtime(TRUE);
111:
112:         // update benchmark
113:         $ti_check += $after - $before;
114:
115:         // print word if misspelled
116:         if ($misspelled)
117:         {
118:             print("$word\n");
119:             $misspellings++;
120:         }
121:
122:         // prepare for next word
123:         $index = 0; $word = "";
124:     }
125: }
126:
127: // close file
128: fclose($fp);
129:
130: // determine dictionary's size
131: $before = microtime(TRUE);
132: $n = size();
133: $after = microtime(TRUE);
134:
135: // calculate time to determine dictionary's size
136: $ti_size = $after - $before;
137:
138: // unload dictionary
139: $before = microtime(TRUE);
140: $unloaded = unload();
141: $after = microtime(TRUE);

```

```
142:
143:     // abort if dictionary not unloaded
144:     if (!$unloaded)
145:     {
146:         print("Could not load $dict.\n");
147:         return 5;
148:     }
149:     // calculate time to determine dictionary's size
150:     $ti_unload = $after - $before;
151:
152:     // report benchmarks
153:     printf("\nWORDS MISPELLED:      %d\n", $mispellings);
154:     printf("WORDS IN DICTIONARY:  %d\n", $n);
155:     printf("WORDS IN FILE:         %d\n", $words);
156:     printf("TIME IN load:           %f\n", $ti_load);
157:     printf("TIME IN check:          %f\n", $ti_check);
158:     printf("TIME IN size:           %f\n", $ti_size);
159:     printf("TIME IN unload:         %f\n", $ti_unload);
160:     printf("TOTAL TIME:             %f\n", $ti_load + $ti_check + $ti_size + $ti_unload);
161:
162: ?>
```