jQuery and JavaScript

PHRASEBOOK
This page intentionally left blank
jQuery and JavaScript

PHRASEBOOK

Brad Dayley
Dedication

For D!
A & F
Contents

1 Jumping into jQuery, JavaScript, and the World of Dynamic Web Development 1
   Understanding JavaScript 2
   Introducing jQuery 4
   Introducing jQuery UI 7
   Introducing jQuery Mobile 9
   Configuring Browser Development Tools 12

2 Using the JavaScript Language 15
   JavaScript Syntax 15
   Defining and Accessing Data 16
   Defining Functions 20
   Manipulating Strings 21
   Manipulating Arrays 25
   Applying Logic 29
   Math Operations 31
   Working with Dates 36

3 Interacting with the Browser 43
   Writing to the JavaScript Console 43
   Reloading the Web Page 44
   Redirecting the Web Page 44
   Getting the Screen Size 45
   Getting Current Location Details 45
   Accessing the Browser 47
   Using the Browser History to Go Forward and Backward Pages 49
   Creating Popup Windows 50
   Manipulating Cookies 52
   Adding Timers 55
4  Accessing HTML Elements  59
    Finding HTML Elements in JavaScript  59
    Using the jQuery Selector to Find HTML Elements  61
    Chaining jQuery Object Operations  75
    Navigating jQuery Objects to Select Elements  76

5  Manipulating the jQuery Object Set  83
    Getting DOM Objects from a jQuery Object Set  84
    Converting DOM Objects into jQuery Objects  84
    Iterating Through the jQuery Object Set  
       Using .each()  85
       Using .map()  87
    Assigning Data Values to Objects  89
    Adding DOM Elements to the jQuery Object Set  91
    Removing Objects from the jQuery Object Set  91
    Filtering the jQuery Object Results  92

6  Capturing and Using Browser and User Events  95
    Understanding Events  96
    Adding Event Handlers  99
    Controlling Events  107
    Using Event Objects  111
    Handling Mouse Events  115
    Handling Keyboard Events  118
    Form Events  122

7  Manipulating Web Page Elements Dynamically  125
    Getting and Setting DOM Element Attributes and Properties  126
Contents

Getting and Setting CSS Properties 130
Getting and Manipulating Element Content 139

8 Manipulating Web Page Layout Dynamically 143
Hiding and Showing Elements 143
Adjusting Opacity 146
Resizing Elements 149
Repositioning Elements 152
Stacking Elements 156

9 Dynamically Working with Form Elements 159
Getting and Setting Text Input Values 160
Checking and Changing Check Box State 161
Getting and Setting the Selected Option in a Radio Group 162
Getting and Setting Select Values 164
Getting and Setting Hidden Form Attributes 166
Disabling Form Elements 167
Showing/Hiding Form Elements 170
Forcing Focus to and Away from Form Elements 172
Controlling Form Submission 175

10 Building Web Page Content Dynamically 177
Creating HTML Elements Using jQuery 178
Adding Elements to the Other Elements 179
Removing Elements from the Page 184
Dynamically Creating a Select Form Element 186
Appending Rows to a Table 189
Inserting Items into a List 191
Creating a Dynamic Image Gallery 193
Adding HTML5 Canvas Graphics 196
11 Adding jQuery UI Elements 201
  Adding the jQuery UI Library 201
  Implementing an Autocomplete Input 203
  Implementing Drag and Drop 205
  Adding Datepicker Element 212
  Using Sliders to Manipulate Elements 215
  Creating a Menu 219
  Adding Tooltips 223

12 Animation and Other Special Effects 227
  Understanding jQuery Animation 228
  Animating Visibility 234
  Making an Element Slide Back to Disappear 238
  Animating Show and Hide 242
  Animating Resizing an Image 246
  Animating Moving an Element 248

13 Using AJAX to Communicate with Web Servers and Web Services 251
  Understanding AJAX 251
  AJAX from JavaScript 261
  AJAX from jQuery 267
  Handling jQuery AJAX Responses 282
  Using Advanced jQuery AJAX 285

14 Implementing Mobile Web Sites with jQuery 291
  Getting Started with jQuery Mobile 291
  Building Mobile Pages 302
  Implementing Mobile Sites with Multiple Pages 306
  Creating a Navbar 314
  Applying a Grid Layout 316
  Implementing Listviews 320
Contents

Using Collapsible Blocks and Sets 326
Adding Auxiliary Content to Panels 327
Working with Popups 329
Building Mobile-Friendly Tables 332
Creating Mobile Forms 334

Index 341
Acknowledgments

I’d like to take this page to thank all those who made this title possible. First, I thank my wonderful wife and boys for giving me the inspiration and support I need. I’d never make it far without you. Thanks to Mark Taber for getting this title rolling in the right direction; Karen Gill for turning the ramblings of my techie mind into coherent text; Phil Ballard for ensuring the accuracy in the book and keeping me honest; Kathy Ruiz and Gloria Schurick for making sure the book is the highest quality; Larry Sweazy for making sure that the readers can actually find what they look for in the book; Tammy Graham and Laura Robbins for their graphical genius; Chuti Prasertsith for the stylish and sleek cover; and Katherine Matejka for all her hard work in making sure this book is the best it can be. You guys are awesome!
About the Author

Brad Dayley is a senior software engineer with 20 years of experience developing enterprise applications. He has used HTML/CSS, JavaScript, and jQuery extensively to develop a wide array of web pages ranging from enterprise application interfaces to sophisticated rich Internet applications to smart interfaces for mobile web services. He is the author of *Python Developer’s Phrasebook* and *Sams Teach Yourself jQuery and JavaScript in 24 Hours*. 
We Want to Hear from You!

As the reader of this book, you are our most important critic and commentator. We value your opinion and want to know what we’re doing right, what we could do better, what areas you’d like to see us publish in, and any other words of wisdom you’re willing to pass our way.

You can email or write directly to let us know what you did or didn’t like about this book—as well as what we can do to make our books stronger.

Please note that we cannot help you with technical problems related to the topic of this book, and that due to the high volume of mail we receive, we might not be able to reply to every message.

When you write, please be sure to include this book’s title and author as well as your name and contact information.

Email: feedback@developers-library.info

Mail: Reader Feedback
Addison-Wesley Developer’s Library
800 East 96th Street
Indianapolis, IN 46240 USA

Reader Services

Visit our Web site and register this book at informit.com/register for convenient access to any updates, downloads, or errata that might be available for this book.
This page intentionally left blank
This page intentionally left blank
3

Interacting with the Browser

Dynamic web pages often require you to access and in some cases even manipulate things beyond the HTML elements. JavaScript provides a rich set of objects and functions that allow you to access information about the screen, browser window, history, and more.

The phrases in this chapter describe ways to use the `screen`, `window`, `location`, and `history` objects that provide JavaScript with an interface to access information beyond the web page. Additional phrases describe utilizing those objects to implement cookies, popup windows, and timers.

Writing to the JavaScript Console

```javascript
console.log("This is Debug"); // "This is Debug" is displayed in console
var x=5;
console.log("x="+x); // "x=5" is displayed in console
```
The JavaScript console can be an important tool when debugging problems in your jQuery and JavaScript code. The console log is simply a location where you can view data output from the JavaScript code. Each of the major browsers has a console log tool that displays the output.

To output data to the console log, use `console.write(DEBUG_STRING)` and pass it the text that you want to display on the console.

### Reloading the Web Page

**location.reload()**

A useful feature of JavaScript is the ability to force the browser to reload the current web page. You may want to reload the web page because data has changed on the server or a certain amount of time has elapsed. The `location.reload()` method requests that the browser reload the current URL.

### Redirecting the Web Page

```
location.href="newpage.html";
location.href="http://jqueryin24/index.html;
```

Another extremely useful feature of JavaScript is the ability to redirect the browser away from the current URL to another location. You do this by setting the `location.href` value to a new URL. The new URL can be a full address, such as `http://mysite.com/newlocation/index.html`, or a relative location to the current URL, such as `page2.html`. 
Getting the Screen Size

```javascript
screen.availHeight; // returns screen height in pixels
screen.availWidth; // returns screen width in pixels
```

An important feature of JavaScript these days is the ability to get the screen size. The screen sizes of browsers vary so much that you often need to use different sets of code for larger, medium, or smaller screens. To get the screen size, use the `screen.availHeight` and `screen.availWidth` attributes. These values are specified in the number of pixels.

Getting Current Location Details

The JavaScript location object provides an easy way to get various pieces of information about the URL the user is currently viewing. Because JavaScript and jQuery code are often used across multiple pages, the location object is the means to get information about what URL the browser is currently viewing. The following sections provide some phrases that provide information about the browser’s current location.

Finding the Current Hash

```javascript
location.hash
```

The `location.hash` value returns the current hash, if any, that the browser is using. This is useful when you are displaying a web page with multiple hash anchors. The hash provides a context to the location on the page the user actually clicked.
Getting the Host Name

location.hostname

The location.hostname value returns the domain name of the server that sent the page to the user. This is just the portion after the protocol but before the port number or path. This allows you to determine which server to send AJAX requests back to. Often, multiple servers may handle the web site, but you may want to interact only with the server that served the web page in the first place.

Looking at the File Path

location.pathname

The location.pathname returns the path that the page was loaded from on the server. The pathname also provides a context as to what page the user is looking at.

Getting the Query String

location.search

The location.search value returns the query string that was passed to the server to load the page. Typically, you think about a query string as a server-side script tool. However, it can be just as valuable to JavaScript code that manipulates the data returned from the server and requests additional information from the server via AJAX.
Determining If the Page Is Viewed on a Secure Protocol

`location.protocol`

The easiest way to determine if a page is being viewed from a secured location on the server is to look at the `location.protocol` value. This value will be http on regular requests or https on secure requests.

Accessing the Browser

Another important object built into JavaScript is the `window` object. The `window` object represents the browser and provides you with a wealth of information about the browser position, size, and much more. It also allows you to open new child windows, close windows, and even resize the window.

Getting the Max Viewable Web Page Size

`window.innerHeight; // returns browser view port height in pixels`
`window.innerWidth; // returns browser view port width in pixels`

The `window` object provides the `innerHeight` and `innerWidth` of the browser window. These values represent the actual pixels in the browser window that the web page will be displayed within. This is a critical piece of information if you need to adjust the size and location of elements on the web page based on the actual area that is being displayed.
Setting the Text Displayed in the Browser Status Bar

```
window.status = "Requesting Data From Server . . .";
```

The browser has a status bar at the bottom. You can set the text that is displayed there to provide to the user additional information that does not belong on the page, such as the server name, current status of requests, and more. To set the text displayed in the browser status bar, set the `window.status` value equal to the string you want displayed.

Getting the Current Location in the Web Page

```
window.pageXOffset;
// number of pixels the page has scrolled to the right
window.pageYOffset;
// returns number of pixels the page has scrolled down
```

When writing dynamic code, it is often necessary to determine the exact location in the web page that is currently being viewed. When the user scrolls down or to the right, the position of the page to the frame of the browser view port changes.

To determine the number of pixels the page has scrolled to the right, use the `window.pageXOffset` attribute. To determine the number of pixels the page has scrolled down, use the `window.pageYOffset` attribute.
Opening and Closing Windows

```javascript
// Opens a new blank window, writes to it, and then closes it.
var newWindow = window.open();
newWindow.document.write("Hello From a New Window");
newWindow.close();
// Opens another URL in a new window
window.open("http://google.com");
```

The window object also provides a set of methods that allow you to create and manage additional child windows from your JavaScript code.

For example, the `window.open(URL)` method opens a new window and returns a new window object. If you do not specify a URL, the browser opens a blank page that can be written to using the `window.document` object.

You can call `.close()` on window objects that you have created, and they will be closed.

Using the Browser History to Go Forward and Backward Pages

```javascript
history.forward(); // forward 1 page
history.back(); // backward 1 page
history.go(-2); // backward 2 pages
```

The browser keeps track of the pages that have been navigated to in a history. JavaScript allows you to access this history to go forward or backward pages. This allows you to provide forward and backward controls to your web pages. You can also use this feature to provide bread crumbs displaying links to multiple pages back in the history.
To go forward one page, use `history.forward()`. To go backward one page, use `history.back()`.

To go forward or backward multiple pages, use `history.go(n)`, where `n` is the number of pages. A negative number goes backward that many pages, and a positive number goes forward that many pages.

### Creating Popup Windows

```javascript
var result = confirm("You Entered " + response + "is that OK?");
if(result){ alert("You Said Yes.") } else {alert("You Said no.")}
```

Window objects provides several different methods that allow you to launch popup windows that you can interact with for alerts, prompts, and notifications. The popup windows are displayed, and the user needs to interact with the popup before continuing to access the web page.

There are three kinds of popup boxes that can be created:

- **alert(msg)** — Launches a popup window that displays an alert message and provides a button to close the popup.

- **confirm(msg)** — Launches a popup window that displays a confirmation and message provides an OK and a Cancel button, which both close the popup. If you click the OK button, the return value from `confirm()` is true; otherwise, it is false.
- **prompt(msg)**—Launches a popup window that displays the message, a text box for input, and an OK and Cancel button, which both close the popup. If you click the OK button, the return value from `prompt()` is the text typed into the text box; otherwise, it is false.

The code that follows illustrates these popup boxes, as shown in Figure 3.1.

```html
01 <html>
02 <head>
03   <title>Python Phrasebook</title>
04   <meta charset="utf-8" /> 
05   <script type="text/javascript"
06     src="../js/jquery-2.0.3.min.js"></script>
07   <script>
08     var response = prompt("What is the airspeed "+
09       "velocity of an unladen swallow:"");
10     var result = confirm("You Entered " +
11       response + 
12       "is that OK?");
13     if(result){ alert("You may pass."") }
14     else {alert("None Shall Pass.")}
15   </script>
16 </head>
17 <body>
18 </body>
```

ch0301.html
By the way

It is often much better to create a fixed position `<div>` element with an overlay than to use these popup boxes because you have much more control over them. I’ll show you how to do just that a little later in the book.

Manipulating Cookies

A common task in JavaScript is getting and setting cookies in the browser. Cookies allow you to store simple key value pairs of information in the browser in a persistent manner. You can access the cookies by the
server-side scripts and JavaScript to determine what those values are.

You can access cookie information using the `document.cookie` object. This object contains all the cookies in the string format `name=value; name=value;` ...

### Setting a Cookie Value in the Browser

```javascript
function setCookie(name, value, days) {
    var date = new Date();
    date.setTime(date.getTime()+(days*24*60*60*1000));
    var expires = ""; expires=""+date.toGMTString();
    document.cookie = name + "=" + value + 
    expires + "; path=/";
}
```

To add a new cookie for your web site, set `document.cookie = "name=value; expireDate; path;";`. The expire date needs to be a date set using `.toGMTString()`, and the path is the path on your web site that the cookie applies to.

### Getting a Cookie Value from the Browser

```javascript
function getCookie(name) {
    var cArr = document.cookie.split(';');
    for(var i=0;i < cArr.length;i++) {
        var cookie = cArr[i].split("=",2);
        cookie[0] = cookie[0].replace(/\s+/,"");
        if (cookie[0] == name){ return cookie; }
    }
}
```

To get the value of the cookie, split the `document.cookie` value using the `;` character, and then iterate
through the resulting array until you find the name you are looking for.

Example Getting and Setting Cookies

The following code shows a full example of setting and getting cookies. When the code is run, two cookies are set: one for name and the other for language. The cookies are then retrieved from the browser and written to the web page, as shown in Figure 3.2.

```html
01 <html>
02 <head>
03   <title>Python Phrasebook</title>
04   <meta charset="utf-8" />
05   <script type="text/javascript" 
06       src="../js/jquery-2.0.3.min.js"></script>
07   <script>
08     function setCookie(name, value, days) {
09       var date = new Date();
10       date.setTime(date.getTime()+(days*24*60*60*
➥1000));
11       var expires = "; expires="+date.toGMTString
➥();
12       document.cookie = name + ";" + value +
13           expires + "; path=/";
14     }
15     function getCookie(name) {
16       var cArr = document.cookie.split(';');
17       for(var i=0;i < cArr.length;i++) {
18         var cookie = cArr[i].split("=",2);
19         cookie[0] = cookie[0].replace(/\s+/,"");
20         if (cookie[0] == name){ return cookie; }
21       }
22   }
```

CHAPTER 3 Interacting with the Browser
Adding Timers

Another useful feature of JavaScript is the ability to set timers that execute a function or evaluate an expression after a certain amount of time or on a specific interval.

Using timers allows you to delay the execution of code so that it does not need to happen at the exact moment an event is triggered or the page is loaded. The following phrases show you how to create timers to delay code execution and to apply recurring actions.
Adding a Delay Timer

```javascript
function myTimer () {
    alert("Timer Function Executed");
}
var timerId = setTimeout(myTimer, 10000);
```

To simply delay the execution of code for a certain amount of time, use the `setTimeout(code, ms)` method, where `code` is either a statement or a function that executes when the time expires. `ms` is the number of milliseconds. For example, to execute a function named `myTimer()` in 10 seconds, you would use the following:

```javascript
setTimeout(myTimer, 10000);
```

Cancel a Timer

```javascript
function myTimer () {
    alert("Timer Function Executed");
}
var timerId = setTimeout(myTimer, 10000);
clearTimeout(timerId); //timer will not execute
```

At any point before the time runs out and the code is executed, you can clear the timer by calling the `clearTimeout(id)` method using the ID returned from `setTimeout()`. Once the timer has been cleared, it does not execute. You need to keep track of the ID returned from `setTimeout()` using a variable.

Adding a Recurring Timer

```javascript
var statusTimerId;
var status = "OK";
//checks status every minute as long as it is "OK"
function checkStatus () {
    if(status == "OK"){
```
You can also start a timer that triggers on a regular interval using the `setInterval(function, ms)` method. This method also accepts a function name and milliseconds as arguments. Inside the function, you need to call `setInterval` again on the same function so that the code will be called again.

To turn off the recurring timer, simply do not call `setInterval()` again inside the timer function, or use `clearInterval()` outside the function.

```javascript
alert("Status OK");
statusTimerId = setInterval(checkStatus, 60000);
} else {
    alert("Status Failed");
}
statusTimerId = setInterval(checkStatus, 60000);
```
This page intentionally left blank
Index

A

absolute values, calculating, 34
accessing
  browsers, 47-49
    accessing cookies, 52-55
  history, 49
HTML
  chaining object operations, 75-76
  elements, 59
  navigating objects, 76-82
  searching elements, 59-68, 71-74
jQuery in JavaScript, 6
libraries, 7
Mobile, 10
variables, 16
addEventListener() function, 103
adding
  auxiliary content to panels, 328
  borders, 134
  conditional blocks of code, 30
  cookies, 53
dividers to lists, 324
DOM elements to objects, 91
elements to elements, 179-183
event handlers, 99, 105
forms
  elements, 336
    mobile, 334-336, 340
initialization code, 100
items to lists, 191-193
JavaScript
  to HTML documents, 3
    loading from external files, 4
jQuery to web pages, 5-6
labels, 335
mouse-click-handling code, 115
navigation buttons, 307, 311, 314
page load event handlers, 99
rows to tables, 189-191
tables, 333
timers, 55-57
transitions, 308
UI elements
   applying sliders, 215-216, 219
   attaching datepicker, 212-215
   coding tooltips, 223-225
   creating menus, 220-222
   downloading libraries, 201-202
dragging/dropping, 205-212
   implementing autocomplete, 203-205
   jQuery, 201
adjusting opacity, 146-149.
   See also modifying
adjValues() function, 197
AJAX
   asynchronous communication, 253-254
cross-domain requests, 254
GET/POST requests, 255
JavaScript, 261-267
jQuery, 267-289
overview of, 251-252
response data types, 256-259
ancestors, searching, 80
.animate() method, 228-229
animation, 227-228
   .hide() method, 242
   .show() method, 243
   .toggle() method, 243-245
CSS settings, 228-229
delaying, 233
images
   moving elements, 248-250
   resizing, 246-248
   queues, 231
   sliding toggles, 239-242
   stopping, 232-233
   visibility, 234-238
appending
   bottom of element’s content, 180
   elements, 141
   rows to tables, 189-191
   text, 140
applications, Mobile, 9-12
applying
   AJAX, 251-252
   asynchronous communication, 253-254
cross-domain requests, 254
GET/POST requests, 255
JavaScript, 261-267
jQuery, 267-289
overview of, 251-252
response data types, 256-259
event objects, 111-114
filters to selectors, 74
for() loops, 31
grid layouts, 316-317, 320
logic, 29-31
.map() method, 87-88
nested lists, 322
popups, 329, 332
power functions, 35
right-click, 117-118
selectors, 62
sliders, 215-216, 219
trigonometric functions, 35
while() loops, 30

arguments, 20

arrays
combining, 26
creating, 18
items
  deleting, 27
  detecting, 27
iterating, 31
manipulating, 25-27
sorting, 28
splicing, 26

assigning
data values to objects, 89
event handlers in HTML, 101

asynchronous communication, 253-254
attaching datepicker element, 212-215
.attr() method, 126, 166
attributes, selecting based on HTML, 64
autocomplete, implementing, 203-205

B
back buttons, creating, 309
behavior, events, 107-110
blocks, adding code, 30
.blur() method, 173
borders, adding, 134
bottom of element’s content, appending, 180
browsers. See also interfaces
accessing, 47-49
cookies, modifying, 52-55
current location details, 45-47
development tools, configuring, 12-13
events
  adding event handlers, 99-106
  forms, 122-123
  keyboards, 118-120
  managing, 107-110
  mouse, 115-118
  objects, 111-114
  overview of, 96
  types, 96, 99
history, accessing, 49
JavaScript consoles, applying, 44
navigating, 43
popup windows, creating, 50-51
screens, sizing, 45
timers, adding, 55, 57
web pages
redirecting, 44
reloading, 44
building mobile pages, 302-304
buttonImage option, 213
buttonImageOnly option, 213
buttons
back, creating, 309
navigation
adding, 307, 311, 314
positioning, 308
split lists, 324
check box state, modifying, 161-162
checking
items in arrays, 27
for substrings, 25
children, retrieving elements, 77
Chrome, enabling
JavaScript in, 13
classes
deleting, 137
names, searching DOM objects, 60
toggling, 137, 139
click() method, 110
closest elements, retrieving, 77
closing windows, 49
code
blocks, adding, 30
dynamic programming.
See dynamic programming
initialization, adding, 100
JavaScript
adding to HTML documents, 3
consoles, 44
loading from external files, 4
overview of, 2
mouse-click-handling, adding, 115
collapsible elements, dividing content into, 326-327
colors, modifying, 131-132
combining
arrays, 26
strings, 23
communication,
asynchronous, 253-254
complete function, 229-230
components, dates, 39
conditional blocks of code,
adding, 30
configuring
browser development
tools, 12-13
cookie values, 53
CSS properties, 130-139
default mobile
settings, 301
DOM element
properties, 126-129
hidden form
attributes, 166
select inputs, 164-165
selected option in radio
groups, 162
text
input values, 160
status bars, 48
timers, 55-57
consoles, JavaScript, 44
content
auxiliary, adding to
panels, 328
HTML elements, select-
ing based on, 66
parent, appending
elements, 141
web pages
adding elements,
179-183
 appending rows to
tables, 189-191
 building dynamically,
177
deleting elements,
184-185
HTML elements, 178
HTML5 canvas
graphics, 197-199
image galleries,
193-196
inserting items into
lists, 191-193
select form
elements, 186-188
Content Discovery
Networks. See CDNs
content tooltip, 223
converting
DOM objects into
jQuery objects, 84
numbers to strings, 22
strings to numbers, 23
coordinates, getting
mouse, 115
cross-domain requests,
AJAX, 254
.css() method, 130
CSSs (Cascading Style
Sheets), 7
animating, 228-229
elements, 130-139
theme swatches, 295
current date and time, getting, 37
current hashes, searching, 45
current location details, 45-47
current location of web pages, 48
customizing. See also configuring
   forms, 334-336, 340
   popups, 329-332
   tables, 333
   
   .data() method, 166
data types, AJAX, 256-259
Date object, 36-40
dateFormat option, 213
datepicker element, attaching, 212-215
dates, components, 39
dblclick() method, 110
debugging JavaScript consoles, 44
default behavior, stopping, 109
default mobile settings, configuring, 301
defining
   functions, 20
   variables, 16
   viewport meta tags, 300
delay timers, adding, 56
delaying animation, 233
deleting
   classes, 137
   elements to elements, 184-185
   event handlers
      JavaScript, 104
      jQuery, 106
   items from arrays, 27
   objects, 91
deltas, formatting time, 39
descendent elements, searching, 78
detecting
   changes to text, 119
   items in arrays, 27
   mobile screen size, 294
Developer Tools (Internet Explorer), 13
development
   browser tools, configuring, 12-13
   jQuery Mobile, 9-12
devices, Mobile, 9-12
disabling form elements, 167-169, 335
dividers, adding to lists, 324
dividing content into collapsible elements, 326-327
Document Object Model. See DOM
documents, adding
   JavaScript to HTML, 3
DOM (Document Object Model), 5
  elements
    adding, 91
    configuring properties, 126-129
    content, 139-141
  objects, 60-61, 84
downloading jQuery UI libraries, 201-202
dragging/dropping elements, 205-212
droppable widget options, 208
duration function, 229-230
dynamic programming
  forms
    check box state, 161-162
    disabling elements, 167-169
    elements, 159
    forcing focus to/away from elements, 172-174
    hidden attributes, 166
    managing submissions, 175
    radio inputs, 162
    select values, 164-165
    showing/hiding elements, 170-172
    text input values, 160
  web pages, 177
    adding elements, 179-183
    adjusting opacity, 146-149
    appending rows to tables, 189-191
    CSS properties, 130-139
    deleting elements, 184-185
    DOM element properties, 126-129
    element content, 139-141
    hiding/viewing elements, 144-146
    HTML elements, 178
    HTML5 canvas graphics, 197-199
    image galleries, 193-196
    inserting items into lists, 191-193
    modifying, 125
    modifying layouts, 143
    repositioning elements, 152-153, 156
    resizing elements, 149, 152
    select form elements, 186-188
    stacking elements, 156-158
elements

appending, 141
children, retrieving, 77
content, 139-141
CSS properties, 130-139
descendent, searching, 78
DOM, 5
adding, 91
configuring properties, 126-129
fading, 239, 242
forms, 159
adding, 336
check box state, 161-162
creating select, 186-188
disabling, 167-169, 335
forcing focus to/away from elements, 172-174
hidden attributes, 166
managing submissions, 175
radio inputs, 162
refreshing, 336
select values, 164-165
showing/hiding, 170-172
text input values, 160

HTML

adding, 179-183
deleting, 184-185
jQuery, 178
moving, animating, 248-250

UIs

adding, 201
applying sliders, 215-219
attaching datepicker, 212-215
coding tooltips, 223-225
creating menus, 220-222
downloading libraries, 201-202
dragging/dropping, 205-212
implementing autocomplete, 203-205
web pages
  hiding/viewing, 144-146
  loading HTML into, 269-271
  repositioning, 152-156
  resizing, 149-152
  stacking, 156-158

enabling
  Developer Tools on Internet Explorer, 13
  JavaScript in Chrome, 13

.eq (index) filter, 92
equality, objects, 29

event handlers
  global, 285
  swipe, 310, 314

 event.preventDefault() method, 109
 event.stopPropagation() method, 109

events
  browsers
    adding event handlers, 99-106
    forms, 122-123
    keyboards, 118-120
    managing, 107-110
    mouse, 115-118
    objects, 111-114
    overview of, 96
    types, 96, 99
    draggable widget, 207
droppable widget, 209

mobile, 295-297
reset, 175
submit, 175

existingObject, 180

external files, loading from JavaScript, 4

fading
  animation elements in/out, 234-235
elements, 239-242
to levels of opacity, 236

files
  accessing, 6
  JavaScript, loading from external, 4
  paths, viewing, 46
  web pages, loading, 5-6

.filter(filter) method, 92

filters
  object results, 92-94
  selectors, applying, 74

finding. See searching
Firefox, installing Firebug, 13

.focus() method, 93

focus, modifying, 122

.fonts() method, 173

fonts, modifying, 136

footers, mobile web pages, 304

for() loops, 31

forcing focus to/away from form elements, 172-174
formatting
arrays, 18
back buttons, 309
dates, 36-40
HTML
adding elements, 179-183
deleting elements, 184-185
elements, 178
image galleries, 193-196
listviews, 320-325
menus, 220-222
mobile pages, 302-304
navbars, 314
objects, 19
popup windows, 50-51
strings, modifying, 21-25
text, input values, 160
time
deltas, 39
strings, 38
tooltips, 223-225
forms
elements, 159
adding, 336
check box state, 161-162
creating select, 186-188
disabling, 167-169, 335
forcing focus
to/away from
elements, 172-174
hidden attributes, 166
managing submissions, 175
radio inputs, 162
refreshing, 336
select values, 164-165
showing/hidden, 170-172
text input values, 160
events, 122-123
HTML elements, selecting based on, 71
mobile, adding, 334-336, 340
functionality, AJAX, 285-289
functions
addEventListener(), 103
adjValues(), 197
calling, 20
complete, 229-230
defining, 20
duration, 229-230
easing, 229-230
handler, 105
power, applying, 35
queue, 230
removeEventListener() function, 104
renderSpark(), 197
trigonometric, applying, 35
G

- galleries, formatting image, 193-196
- generating random numbers, 32
- GET requests, AJAX, 255, 262-264
- `.get()` method, 84
  - JSON, handling, 271-274
  - XML, handling, 274, 277
- `.getScript()` method, 256
- getting
  - CSS properties, 130-139
  - DOM element properties, 126-129
  - Elements, content, 139-141
  - event target objects, 114
  - hidden form attributes, 166
  - mouse coordinates, 115
  - select inputs, 164-165
  - selected option in radio groups, 162
  - text input values, 160
- global event handlers, 285
- global setup, modifying, 285
- graphics, HTML5 canvas, 197-199
- grids, layouts, 316-320
- groups, getting/setting radio options, 162

H

- handlers
  - functions, 105
  - events
    - adding, 99
    - swipe, 310-314
  - global event, 285
- handling
  - events
    - forms, 122-123
    - keyboards, 118-120
    - mouse, 115-118
    - JSON data, 271-274
    - selection changes, 123
    - text data (AJAX), 257
    - XML data, 274-277
- `.has(selector or element)` method, 93
- hashes, current, 45
- headers, mobile web pages, 304
- hidden form attributes, 166
- `.hide()` method, 144, 170, 242
- hiding
  - elements, web pages, 144-146
  - form elements, 170-172
  - labels, 335
- hierarchies, selecting based on HTML elements, 68
- history, navigating browsers, 49
hosts, searching names, 46

HTML (Hypertext Markup Language)
  elements
    accessing, 59
    adding, 179-183
    appending/prepending text, 140
  chaining object operations, 75-76
  deleting, 184-185
  getting content of, 140
  jQuery, 178
  navigating objects, 76-82
  repositioning, 152-156
  resizing, 149, 152
  searching, 59-68, 71-74
  stacking, 156-158
  event handlers,
    assigning in, 101
    JavaScript, adding to, 3
    response data, 259

HTML5 canvas graphics, 196-199
  .html() method, 140

resizing, animating, 246-248
source files, modifying, 128
transitions, adding, 237-238

implementing
  autocomplete, 203-205
  low-level AJAX requests, 287-289
  mobile sites with multiple pages, 306-314
  searchable lists, 325
  split button lists, 324

initialization code, adding, 100
innerHeight, 47
innerWidth, 47

input
  autocomplete, implementing, 203-205
  radio, 162
  select, 164-165
  text values, 160

inserting into middle of element’s content, 181

installing Firebug on Firefox, 13

interfaces
  browser development tools, configuring, 12-13
  loading, 9
  navigating, 7

Internet Explorer, enabling Developer Tools, 13

ID, searching DOM objects, 60
images
  galleries, creating, 193-196
is() method, 161-163
items
  arrays
    deleting, 27
detecting, 27
  lists, 191-193
tooltip, 223
iterating
  arrays, 31
  jQuery objects, 85-86
through object properties, 31

J

JavaScript
  AJAX from, 261-267
arrays
    creating, 18
    manipulating, 25-27
Chrome, enabling in, 13
consoles, 44
Date object, 36-40
event handlers
    adding, 103
deleting, 104
events, 96, 99
    adding page load event handlers, 99
    managing, 107-110
external files, loading from, 4
functions, defining, 20
GET requests, sending from, 262-264

HTML
  adding to documents, 3
  searching elements, 59-61
jQuery, accessing, 6
logic, applying, 29-31
math operations, 31-35
objects, creating, 19
on-demand, 255
overview of, 2
POST requests, sending from, 264, 267
strings, manipulating, 21-25
syntax, 15
variables, defining, 16

JavaScript Object Notation. See JSON

.join() method, 27

jQuery
  AJAX from, 267-289
  animation, 227-228
    CSS settings, 228-229
delaying, 233
.hide() method, 242
queues, 231
.show() method, 243
sliding toggles, 239-242
stopping, 232-233
.toggle() method, 243-245
visibility, 234-238
event handlers
  adding, 105
  deleting, 106
events, 96, 99
HTML elements, 178
initialization code, 100
JavaScript, accessing, 6
Mobile, 9-12
mobile web sites, 291
  applying grid layouts, 316-320
building web pages, 302-304
creating navbars, 314
customizing popups, 329, 332
dividing into collapsible elements, 326-327
formatting listviews, 320-325
forms, 334-340
implementing with multiple pages, 306-314
overview of, 291-300
tables, 333
viewing panels, 328
objects
  adding DOM elements to, 91
  applying.map() method, 87-88
  assigning data values to, 89
  chaining operations, 75-76
converting DOM objects into, 84
deleting, 91
filtering results, 92-94
getting, 84
iterating, 85-86
modifying, 83
navigating to select elements, 76-82
overview of, 4
UIs
  accessing libraries, 7
  adding, 201
  applying sliders, 215-219
  attaching datepicker, 212-215
coding tooltips, 223-225
creating menus, 220-222
downloading libraries, 201-202
dragging/dropping, 205-212
implementing autocomplete, 203-205
loading, 9
navigating, 7
web pages, loading, 5-6
jQuery Selector, searching
HTML elements, 61-74
JSON (JavaScript Object Notation)
  handling, 271-274
  response data, 257-258
JSON.parse() method, 258
JSONP (JSON with Padding), 255

K–L

keyboard events, 118-120
keys, pressing, 120
keywords, 17, 96, 99

labels, adding/hiding, 335
languages
  JavaScript
    adding to HTML documents, 3
    loading from external files, 4
    overview of, 2
    syntax, 15
.last() method, 93

layouts
  grids, applying, 316-317, 320
web pages
  adjusting opacity, 146-149
  hiding/viewing elements, 144-146
  modifying, 143
  repositioning elements, 152-153, 156
  resizing elements, 149-152
  stacking elements, 156-158
length, strings, 22
levels, tolerance, 208
libraries
  accessing, 7
  CDNs, loading, 5
  jQuery Mobile, 291
    applying grid layouts, 316-320
    building web pages, 302-304
    creating navbars, 314
    customizing popups, 329-332
    dividing into collapsible elements, 326-327
    formatting listviews, 320-325
    forms, 334-340
    implementing with multiple pages, 306-314
    overview of, 291-300
    tables, 333
    viewing panels, 328
  jQuery UI, adding, 201-202
  loading, 9
  Mobile, 9-12
links, modifying locations, 127
lists
  dividers, adding, 324
  items, inserting into, 191-193
  nesting, 322
searchable, implementing, 325
split button, 324
listviews, formatting, 320-325
.load() method, 101
loading
HTML into page elements, 269-271
JavaScript from external files, 4
jQuery in web pages, 5-6
libraries, 5, 9
Mobile, 12
mobile pages without displaying, 299
.loadPage() method, 298
location.reload() method, 44
locations
current location, 45-47
links, modifying, 127
web pages, 47-48
logic, applying, 29-31
loops
for(), 31
while(), 30
low-level AJAX requests, 287-289

M
managing
events, 107-110
form submissions, 175

manipulating
arrays, 25-27
cookies, 52-55
strings, 21-25
.map() method, applying, 87-88
Math object, 31-35
max option, 216
maximum numbers in sets, 34
menus, formatting, 220-222
meta tags, defining viewports, 300
methods
.animate(), 228-229
.attr(), 126, 166
autocomplete, 203
.blur(), 173
.changePage(), 298
click(), 110
css(), 130
data(), 166
datepicker(), 213
dblclick(), 110
delay(), 234
each(), 85
event.preventDefault(), 109
event.stopPropagation(), 109
event.stopPropagation(), 109
.filter(filter), 92
.first(), 93
.focus(), 173
.get()
  JSON data, 271-274
  XML data, 274, 277
.get([index]), 84
.getScript(), 256
.has(selector or element), 93
.hide(), 144, 170, 242
.html(), 140
.is(), 161-163
.join(), 27
JSON.parse(), 258
.last(), 93
.load(), 101
.loadPage(), 299
location.reload(), 44
.map(), 87-88
.not(filter) method, 94
.off(), 106
.offset(), 153
.on(), 105
onloadHandler(), 100
.pop(), 27
.position(), 153
.post(), 278-281
.prop(), 126, 166
.push(), 18
.ready(), 101
.remove(), 184
.removeAttr(), 162
.setTimeout(), 197
.show(), 144, 170, 243
.sort(), 28
.slice(start, [end]), 94
.slideToggle(), 239
.toggle(), 243, 245
.toLowerCase(), 24
.toUpperCase(), 24
.val(), 160, 164
middle of element’s content, inserting into, 181
minimum numbers in sets, 33
Mobile (jQuery), 9-12
mobile web sites (jQuery), 291
  overview of, 291-300
  web pages
    applying grid layouts, 316-320
    building, 302-304
    creating navbars, 314
    customizing popups, 329-332
    dividing into collapsible elements, 326-327
    formatting listviews, 320-325
    forms, 334-340
    implementing with multiple, 306-314
    tables, 333
    viewing panels, 328
modifying
  check box state, 161-162
  colors, 131-132
  cookies, 52-55
elements
  applying sliders, 215-219
  content, 139-141
focus, 122
fonts, 136
global setup, 285
image source files, 128
link locations, 127
mobile web pages, 298
objects, 83
  adding DOM elements to, 91
  applying.map() method, 87-88
  assigning data values to, 89
  converting DOM objects into, 84
deleting, 91
  filtering results, 92-94
  getting, 84
  iterating, 85-86
radio inputs, 162
select inputs, 164-165
selections, 123
strings, 21-25
text, detecting, 119
web pages
  adjusting opacity, 146-149
  CSS properties, 130-139
  DOM element properties, 126-129
dynamic programming, 125
element content, 139-141
hiding/viewing elements, 144, 146
layouts, 143
repositioning elements, 152-153, 156
resizing elements, 149-152
stacking elements, 156-158
mouse
  coordinates, getting, 115
  events, 115-118
  mouse-click-handling code, adding, 115
  mouseout events, 116
  mouseover events, 116
moving
  elements, animating, 248-250
  HTML elements, 152-156
N
names
  attributes, 166
classes, searching
  DOM objects, 60
hosts, searching, 46
tags, searching DOM objects, 61
navbars, formatting, 314
navigating
browsers, 43
  accessing, 47-49
  adding timers, 55-57
applying JavaScript
  consoles, 44
current location
details, 45-47
history, 49
modifying cookies, 52-55
popup windows, 50-51
redirecting web
  pages, 44
reloading web
  pages, 44
sizing screens, 45
jQuery Mobile library, 291-300
objects to select
elements, 76-82
Uls, 7
  accessing libraries, 7
  loading, 9
navigation buttons
  adding, 307-314
  positioning, 308
nested lists, applying, 322
networks, CDNs, 5
.not(filter) method, 94
numberOfMonths
  option, 213
numbers
  dates, formatting, 36-40
  maximum in sets, 34
  minimum in sets, 33
  random, generating, 32
  rounding, 33
  strings, converting, 22
jQuery
  adding DOM elements to, 91
  applying map() method, 87-88
  assigning data values to, 89
  converting DOM objects into, 84
  deleting, 91
  filtering results, 92-94
  iterating, 85-86
location, 45-47
Math, 31, 34-35
modifying, 83
operations, chaining, 75-76
properties, iterating, 31
strings, special characters, 21
window.XMLHttpRequest, 261
windows, 47-49
XMLHttpRequest, 261
.off() method, 106
.offset() method, 153
.on() method, 105
on-demand JavaScript, 255
onloadHandler() method, 100
onSelect option, 213
opacity
adjusting, 146-149
fading to levels of, 236
opening windows, 49
operations
math, 31-35
objects, chaining, 75-76
options
animation, 229
draggable widget, 205
droppable widget, 208
radio groups,
  getting/setting, 162
orientation options, 216
parents
  content, appending elements, 141
  searching, 80
paths, viewing files, 46
physical events, 107. See also events
pop() method, 27
popups
  creating, 50-51
  applying, 329, 332
  position tooltip, 223
  .position() method, 153
positioning
  HTML elements,
    152-156
  navigation buttons, 308
  parents
    retrieving, 80
    searching, 80
POST requests, AJAX, 255, 264, 267
.pop() method, 278-281
power functions,
  applying, 35
prepending
  content, 179
  text, 140
pressing keys, 120
previous siblings, 81
programming
  dynamic, 125. See also dynamic programming
  JavaScript
    adding to HTML documents, 3
pages
  elements, loading HTML into, 269-271
  load event handlers, adding, 99
panels, viewing, 328
loading from external files, 4
overview of, 2
syntax, 15
.prop() method, 126, 166
propagation, stopping event, 109
properties
CSS, getting/setting, 130-139
DOM, configuring elements, 126-129
objects, iterating, 31
push() method, 18

Q–R
queries, retrieving strings, 46
queues
animating, 231
functions, 230
radio inputs, 162
random numbers, generating, 32
range option, 216
.ready() method, 101
recurring timers, adding, 57
redirecting web pages, 44
refreshing form elements, 336
reloading web pages, 44
.remove() method, 184
removeAttr() method, 162
removeEventListener() function, 104
renderSpark() function, 197

replacing
strings, 25
text, 141
repositioning HTML elements, 152-156
requests, AJAX
cross-domain, 254
GET/POST, 255
JavaScript, 261-267
jQuery, 267-289
response data types, 256-259
reset event, 175
resizing
HTML elements, 149-152
images, animating, 246-248
responses, AJAX
jQuery, 282-284
data types, 256-259
results, filtering objects, 92-94
retrieving, 79
children elements, 77
cookie values, 53
parents, positioning, 80
previous siblings, 81
query strings, 46
siblings, 79-82
right-click, applying, 117-118
rounding numbers, 33
rows, appending tables, 189-191
screens
  mobile, detecting size, 294
  sizing, 45
searchable lists, implementing, 325
searching
  ancestors, 80
  characters, 22
  current hashes, 45
  current location of web pages, 48
  descendent elements, 78
  host names, 46
HTML
  chaining object operations, 75-76
  elements, 59-68, 71-74
  parents, 80
  strings, 25
secure locations, viewing from, 47
select form elements, creating, 186-188
select inputs, 164-165
selecting
  elements, navigating objects to, 76-82
  modifying, 123
selectors
  applying, 62
  filters, 74
sending
  GET requests from JavaScript, 262-264
  POST requests from JavaScript, 264-267
servers
  AJAX
    asynchronous communication, 253-254
    cross-domain requests, 254
    GET/POST requests, 255
    JavaScript, 261-267
    jQuery, 267-289
    overview of, 251-252
    response data types, 256-259
    updating, 278-281
sets
  maximum numbers in, 34
  minimum numbers in, 33
setTimeout() method, 197
settings. See also configuring
  CSS
    animating, 228-229
    properties, 130-139
  DOM element properties, 126-129
  hidden form attributes, 166
  select inputs, 164-165
  selected option in radio groups, 162
  text input values, 160
setup, modifying global, 285
.show() method, 144, 170, 243
showButtonPanel option, 213
showOn option, 213
siblings
  previous, 81
  retrieving, 79-82
sizing
  screens, 45, 294
  web pages, 47
.slice(start, [end])
  method, 94
slide option, 216
.slideToggle() method, 239
sliders, applying, 215-216, 219
sliding toggles, 239-242
sorting arrays, 28
source files, modifying
  images, 128
special characters, 21
special effects, animation, 227-228
  CSS settings, 228-229
  delaying, 233
.hide() method, 242
moving elements, 248-250
queues, 231
resizing images, 246-248
.show() method, 243
sliding toggles, 239-242
stopping, 232-233
.toggle() method, 243-245
visibility, 234-238
splicing
  arrays, 26
  strings, 24
split button lists, 324
splitting strings, 24
stacking HTML elements, 156-158
state, modifying check boxes, 161-162
status bars, configuring
  text, 48
step function, 231
stopping
  animation, 232-233
  default behavior, 109
  event propagation, 109
strings
  case modifying, 24
  combining, 23
  dates, formatting, 37
  manipulating, 21-25
  numbers, converting, 22
  queries, retrieving, 46
  replacing, 25
  searching, 25
  splicing, 24
  splitting, 24
  substrings, checking for, 25
  time, formatting, 38
submit event, 175
submitting forms, managing, 175
substrings, checking for, 25
swatches, theme, 295
swipe event handlers, 310, 314
syntax, JavaScript, 15

tab-separated strings, creating arrays from, 27
tables
  adding, 333
  rows, appending, 189-191
tags, searching DOM objects, 61
targets, getting event objects, 114
text
  AJAX, 257
  appending/ prepending, 140
  autocomplete, implementing, 203-205
  input values, 160
  modifying, detecting, 119
  replacing, 141
  status bars, configuring, 48
theme swatches, 295
time
  current, getting, 37
  deltas, formatting, 39
  strings, formatting, 38
timers, adding, 55-57
.toggle() method, 243-245
toggling
  classes, 137-139
  element visibility on/off, 235
tolerance levels, 208
.toLowerCase() method, 24
tooltips, formatting, 223-225
.toUpperCase() method, 24
transitions
  adding, 308
  animation, 237-238
triggering
  animation, 232
  events manually, 110
trigonometric functions, applying, 35
types of events, 96, 99

U
Uls (user interfaces)
  elements
    adding, 201
    applying sliders, 215-219
    attaching datepicker, 212-215
coding tooltips, 223-225
    creating menus, 220-222
downloading
libraries, 201-202
dragging/dropping,
205-212
implementing auto-
complete, 203-205
libraries, accessing, 7
loading, 9
updating servers, 278-281
user interaction, adding,
179-183
user interfaces. See UIs

V

.val() method, 160, 164
value option, 216
values
absolute, calculating, 34
attributes, 166
cookies, configuring, 53
objects, assigning, 89
text input, 160
var keyword, 17
variables
accessing, 16
defining, 16
viewing
elements, web pages,
144-146
file paths, 46
form elements,
170-172
listviews, formatting,
320-325
mobile pages,
loading without
displaying, 299
panels, 328
web pages, 47
web sites on mobile
devices, 294
viewports, defining meta
tags, 300
visibility
animation, 234-238
HTML elements,
selecting based
on, 72

W

web forms, 159. See also
forms
web pages. See also HTML
content
adding elements,
179-183
appending rows to
tables, 189-191
building dynamically,
177
deleting elements,
184-185
HTML elements, 178
HTML5 canvas
graphics, 197-199
image galleries,
193-196
inserting items into
lists, 191-193
select elements,
186-188
current location of, 48

dynamic programming

CSS properties, 130-139

DOM element properties, 126-129

element content, 139-141

modifying, 125

events

adding event handlers, 99-106

forms, 122-123

keyboards, 118-120

managing, 107-110

mouse, 115-118

objects, 111-114

overview of, 96

types, 96-99

history, navigating, 49

HTML, loading into, 269-271

jQuery, loading, 5-6

layouts

adjusting opacity, 146-149

hiding/viewing elements, 144-146

modifying, 143

repositioning elements, 152-156

resizing elements, 149-152

stacking elements, 156-158

mobile, building, 302-304

navbars, formatting, 314

redirecting, 44

reloading, 44

secured locations, viewing from, 47

UI elements

adding, 201

applying sliders, 215-219

attaching datepicker, 212-215

coding tooltips, 223-225

creating menus, 220-222

downloading libraries, 201-202

dragging/dropping, 205-212

implementing autocomplete, 203-205

viewing, 47

web servers, AJAX

asynchronous communication, 253-254

cross-domain requests, 254

GET/POST requests, 255

JavaScript, 261-267

jQuery, 267-289

overview of, 251-252

response data types, 256-259
web sites, mobile (jQuery), 291. See also web pages
  applying grid layouts, 316-320
  building web pages, 302-304
  creating navbars, 314
  customizing popups, 329-332
  dividing into collapsible elements, 326-327
  formatting listviews, 320-325
  forms, 334-340
  implementing with multiple pages, 306-314
  overview of, 291-300
  tables, 333
  viewing panels, 328

while() loops, 30

widgets
  datepicker, attaching, 212-215
  draggable/droppable, 205-212
  menus, creating, 220-222
  tooltips, creating, 223-225

window.XMLHttpRequest object, 261

windows
  closing, 49
  objects, 47-49
  opening, 49
  popup, creating, 50-51

X–Z

XML (Extensible Markup Language)
  handling, 274, 277
  response data, 259

XMLHttpRequest object, 261

z-index, 157