

A Brain-Friendly Guide

Head First HTML5 Programming



Learn the secrets of
the HTML5 guru

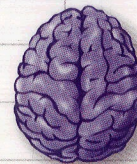


Discover why
everything your
friends know
about video is
probably wrong

Avoid
embarrassing
browser
support issues



A learner's guide
to building web apps
with JavaScript



Load HTML5 and
JavaScript straight
into your brain



Watch out for
common browser
pitfalls



O'REILLY®

Eric Freeman & Elisabeth Robson

Table of Contents (Summary)

	Intro	ix
1	Getting to know HTML5: <i>Welcome to Webville</i>	1
2	Introducing JavaScript and the DOM: <i>A Little Code</i>	35
3	Events, Handlers and All That Jazz: <i>A Little Interactivity</i>	85
4	JavaScript Functions and Objects: <i>Serious JavaScript</i>	113
5	Making Your HTML Location Aware: <i>Geolocation</i>	165
6	Talking to The Web: <i>Extroverted Apps</i>	213
7	Bringing Out Your Inner Artist: <i>Canvas</i>	281
8	Not Your Father's TV: <i>Video... With Special Guest Star "Canvas"</i>	349
9	Storing Things Locally: <i>Web Storage</i>	413
10	Putting JavaScript to Work: <i>Web Workers</i>	473
	Appendix: Top Ten Topics (We Didn't Cover)	531
	Index	549

Table of Contents (the real thing)

Intro

Your brain on HTML5 Programming. Here you are trying to *learn* something, while here your *brain* is doing you a favor by making sure the learning doesn't *stick*. Your brain's thinking, "Better leave room for more important things, like which wild animals to avoid and whether naked snowboarding is a bad idea." So how *do* you trick your brain into thinking that your life depends on knowing HTML5 and JavaScript?

Who is this book for?	xxii
We know what your're thinking	xxiii
And we know what your <i>brain</i> is thinking	xxiii
Metacognition: thinking about thinking	xxv
The technical review team	xxx
Acknowledgments	xxxi

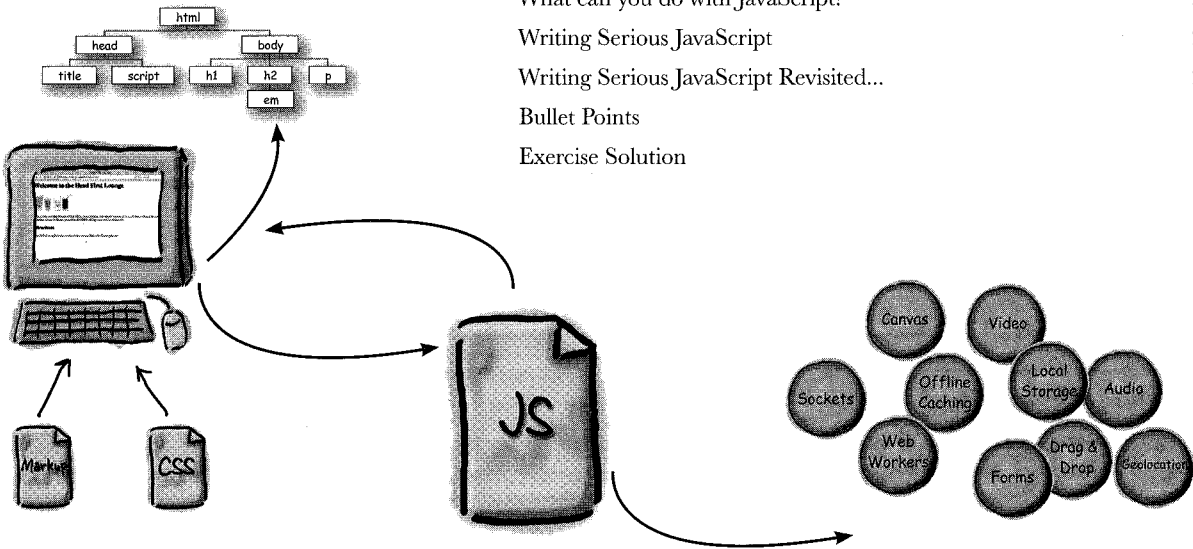
getting to know HTML5

Welcome to Webville

1

HTML has been on a wild ride. Sure, HTML started as a mere markup language, but more recently HTML's put on some major muscle. Now we've got a language tuned for building true web applications with local storage, 2D drawing, offline support, sockets and threads, and more. The story of HTML wasn't always pretty, and it's full of drama (we'll get to all that), but in this chapter, we're first going to go on a quick joyride through Webville to get sense for everything that goes into "HTML5." Come on, hop in, we're headed to Webville, and we're going to start by going from zero to HTML5 in 3.8 pages (flat).

Upgrade to HTML5 TODAY!	2
Introducing the HTML5-o-Matic , update your HTML now!	4
You're closer to HTML5 markup than you think	7
HTML5 Exposed: Confessions of the newest version of HTML	11
Would the REAL HTML5 please stand up...	12
How HTML5 really works...	14
Who Does What?	16
Your First Mission: Browser Reconnaissance	17
What can you do with JavaScript?	22
Writing Serious JavaScript	25
Writing Serious JavaScript Revisited...	26
Bullet Points	31
Exercise Solution	33



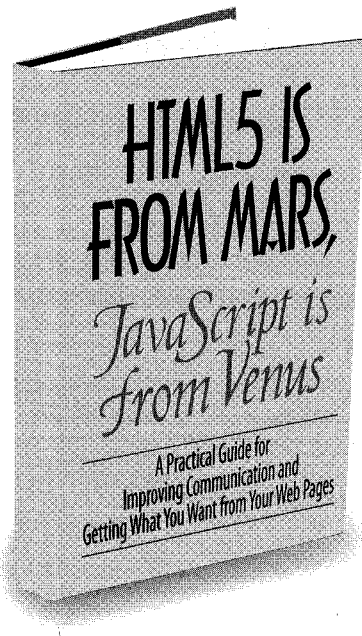
introducing JavaScript and the DOM

2

A Little Code

JavaScript is going to take you to new places. You already know all about HTML markup (otherwise known as *structure*) and you know all about CSS style (otherwise known as *presentation*), but what you've been missing is JavaScript (otherwise known as *behavior*). If all you know about are structure and presentation, sure, you can create some great-looking pages, but they're still *just pages*. When you add behavior with JavaScript, you can create an interactive experience; or, even better, you can create full blown web applications. Get ready to add the most interesting and versatile skill in your web toolkit: JavaScript and programming!

The Way JavaScript Works	36
What can you do with JavaScript?	37
Declaring a variable	38
How to name your variables	40
Getting Expressive	43
Doing things over and over...	46
Make decisions with JavaScript	49
Making more decisions... and, adding a catch all	50
How and where to add JavaScript to your pages	53
How JavaScript interacts with your page	54
How to bake your very own DOM	55
A first taste of the DOM	56
HTML5 is From Mars, JavaScript is from Venus	58
You can't mess with the DOM until the page has fully loaded.	64
So, what else is a DOM good for anyway?	66
Can we talk about JavaScript again?	
Or, how to store multiple values in JavaScript	67
The Phrase-o-Matic	71
Bullet Points	75
Exercise Solutions	77



events, handlers and all that jazz

3

A Little Interactivity

You still haven't reached out to touch your user.

You've learned the basics of JavaScript but can you get interactive with your users? When pages respond to user input, they aren't just documents anymore, they're living, reacting applications. In this chapter you're going to learn how to handle one form of user input (excuse the pun), and wire up an old-fashioned HTML `<form>` element to actual code. It might sound dangerous, but it's also powerful. Strap yourself in, this is a fast moving to-the-point-chapter where we go from zero to interactive app in no time.



Get ready for Webville Tunes	86
Getting started...	87
But nothing happens when I click "Add Song"	88
Handling Events	89
Making a Plan...	90
Getting access to the "Add Song" button	90
Giving the button a click handler	91
A closer look at what just happened...	92
Getting the song name	94
How do we add a song to the page?	97
How to create a new element	99
Adding an element to the DOM	100
Put it all together...	101
... and take it for a test drive	101
Review—what we just did	102
How to add the Ready Bake Code...	105
Integrating your Ready Bake Code	106
Bullet Points	108
Exercise Solutions	110

javascript functions and objects

Serious JavaScript

4

Can you call yourself a scripter yet? Probably—you already know your way around a lot of JavaScript, but who wants to be a scripter when you can be a programmer? It's time to get serious and take it up a notch—it's time you learn about **functions** and **objects**. They're the key to writing code that is more powerful, better organized and more maintainable. They're also heavily used across HTML5 JavaScript APIs, so the better you understand them the faster you can jump into a new API and start ruling with it. Strap in, this chapter is going to require your undivided attention...



Expanding your vocabulary	114
How to add your own functions	115
How a function works	116
Anatomy of a Function	121
Local and Global Variables	123
Knowing the scope of your local and global variables	124
Oh, did we mention functions are also values?	128
Did someone say “Objects”?!	131
How to create an object in JavaScript	132
Some things you can do with objects	133
Let's talk about passing objects to functions	136
Objects can have behavior too...	142
Meanwhile back at Webville Cinema...	143
Adding the “this” keyword	145
How to create a constructor	147
How does <code>this</code> really work?	149
Test drive your constructor right off the factory floor	153
What is the window object anyway?	155
A closer look at <code>window.onload</code>	156
Another look at the document object	157
A closer look at <code>document.getElementById</code>	157
One more object to think about: your element objects	158
Bullet Points	160

making your html location aware

5

Geolocation

Wherever you go, there you are. And sometimes knowing where you are makes all the difference (especially to a web app). In this chapter we're going to show you how to create web pages that are **location aware**—sometimes you'll be able to pin point your users down to the corner they're standing on, and sometimes you'll only be able to determine the area of town they're in (but you'll still know the town!). Heck, sometimes you won't be able to determine anything about their location, which could be for technical reasons, or just because they don't want you being so nosy. Go figure. In any case, in this chapter we're going to explore a JavaScript API: Geolocation. Grab the best location-aware device you have (even if it's your desktop PC), and let's get started.

Location, Location, Location	166
The Lat and Long of it...	167
How the Geolocation API determines your location	168
Just where are you anyway?	172
How it all fits together	176
Revealing our secret location...	179
Writing the code to find the distance	181
How to add a Map to your Page	183
Sticking a Pin in it...	186
The other cool things you can do with the Google Maps API	188
Can we talk about your accuracy?	191
"Wherever you go, there you are"	192
Getting the app started	193
Reworking our old code...	194
Time to get moving!	196
You've got some Options...	198
The world of timeouts and maximum age...	199
Don't Try This at Home (Pushing Geo to the Limit)	202
Let's finish this app!	204
Integrating our new function	205
Bullet Points	207
Exercise Solutions	209



talking to the web

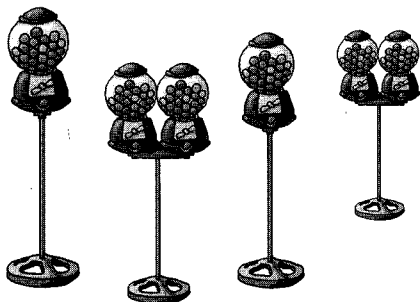
Extroverted Apps**6**

You've been sitting in your page for too long. It's time to get out a little, to talk to web services, to gather data and to bring it all back so you can build better experiences mixing all that great data together. That's a big part of writing modern HTML5 applications, but to do that you've got to *know how* to talk to web services. In this chapter we're going to do just that, and incorporate some data from a real web service right in your page. And, after you've learned how to do that you'll be able to reach out and touch any web service you want. We'll even fill you in on the hippest new lingo you should use when talking to web services. So, come on, you're going to use some more APIs, the communications APIs.



Watch out for the cliffhanger in this chapter!

Mighty Gumball wants a Web app	214
A little more background on Mighty Gumball	216
So how do we make requests to web services?	219
How to make a request from JavaScript	220
Move over XML, meet JSON	226
Writing an onload handler function	229
Displaying the gumball sales data	230
How to set up your own Web Server	231
Reworking our code to make use of JSON	236
Moving to the Live Server	237
It's a cliffhanger!	239
Remember, we left you with a cliffhanger? A bug.	242
What Browser Security Policy?	244
So, what are our options?	247
Meet JSONP	252
But what is the "P" in JSONP for?	253
Let's update the Mighty Gumball web app	256
Step 1: Taking care of the script element...	264
Step 2: Now, it's time for the timer	265
Step 3: Reimplementing JSONP	267
We almost forgot: watch out for the dreaded browser cache	272
How to remove duplicate sales reports	273
Updating the JSON URL to include the lastreporttime	275
Bullet Points	277



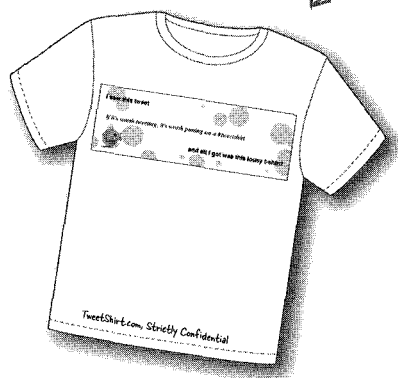
bringing out your inner artist

7

The Canvas

HTML's been liberated from being just a "markup" language. With HTML5's new canvas element you've got the power to create, manipulate and destroy *pixels*, right in your own hands. In this chapter we'll use the canvas element to bring out your inner artist—no more talk about HTML being all semantics and no presentation; with canvas we're going to paint and draw with color. Now it's *all* about presentation. We'll tackle how to place a canvas in your pages, how to draw text and graphics (using JavaScript of course), and even how to handle browsers that don't support the canvas element. And canvas isn't just a one-hit wonder; you're going to be seeing a lot more of canvas in other chapters in this book.

A new HTML5 startup is just waiting
for you to get it off the ground!



Our new startup: TweetShirt	282
Checking out the "comps"	283
How to get a canvas into your web page	286
How to see your canvas	288
Drawing on the Canvas	290
Failing gracefully	295
TweetShirt: the Big Picture	297
First, let's get the HTML in place	300
Now, let's add the <form>	301
Time to get computational, with JavaScript	302
Writing the drawSquare function	304
Add the call to fillBackgroundColor	307
Meanwhile, back at TweetShirt.com...	309
Drawing with Geeks	311
Breaking down the arc method	314
A little taste of using the arc	316
I say degree, you say radian	317
Back to writing the TweetShirt circle code	318
Writing the drawCircle function...	319
Getting your tweets	323
Canvas Text Up Close	328
Giving drawText a spin	330
Completing the drawText function	331
Bullet Points	338
Exercise Solutions	341

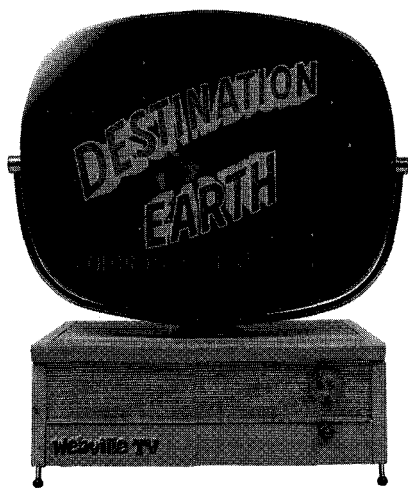


not your father's tv

8

Video... with special guest star “Canvas”

We don't need no plug-in. After all, video is now a first-class member of the HTML family—just throw a `<video>` element in your page and you've got instant video, even across most devices. But video is *far more* than *just an element*, it's also a JavaScript API that allows us to control playback, create our own custom video interfaces and integrate video with the rest of HTML in totally new ways. Speaking of *integration*... remember there's that *video and canvas connection* we've been talking about—you're going to see that putting video and canvas together gives us a powerful new way to *process video* in real time. In this chapter we're going to start by getting video up and running in a page and then we'll put the JavaScript API through its paces. Come on, you're going to be amazed what you can do with a little markup, JavaScript and video & canvas.



↪ Tune in to Webville TV...

Meet Webville TV	350
Plug that set in and test it out...	351
How does the video element work?	353
Closely inspecting the video attributes...	354
What you need to know about video formats	356
How to juggle all those formats...	358
I was told there would be APIs?	363
A little content “programming” on Webville TV	364
How to write the “end of video” handler	367
How the <code>canPlayType</code> method works	369
Unpacking the Demo Unit	375
Inspecting the rest of the factory code	376
The <code>setEffect</code> and <code>setVideo</code> handlers	378
Implementing the video controls	384
Switching test videos	387
It's time for special effects	389
How video processing works	392
How to process video using a scratch buffer	393
Implementing a scratch buffer with Canvas	395
Now we need to write some effects	399
How to use error events	406
Bullet Points	408
Exercise Solutions	410

storing things locally

Web Storage

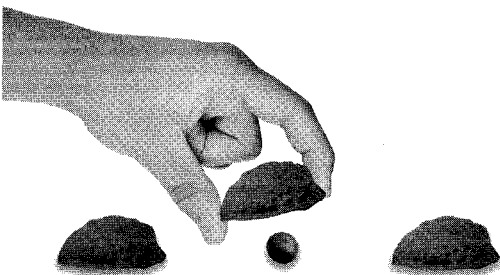
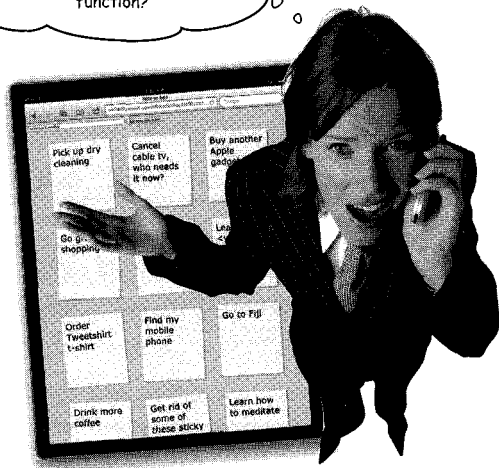
9

Tired of stuffing your client data into that tiny closet

cookie? That was fun in the 90s, but we've got much bigger needs today with web apps. What if we said we could get you five megabytes on every user's browser? You'd probably look at us like we were trying to sell you a bridge in Brooklyn. Well, there's no need to be skeptical—the HTML5 Web storage API does just that! In this chapter we're going to take you through everything you need to store any object locally on your user's device and to make use of it in your web experience.

How browser storage works (1995 - 2010)	414
How HTML5 Web storage works	417
Note to self..	418
Were Local Storage and the Array separated at birth?	424
Creating the interface	429
Now let's add the JavaScript	430
Completing the user interface	431
We need to stop for a little scheduled service	434
Do-It-Yourself maintenance	435
We have the technology	439
Reworking our app to use an array	440
Converting createSticky to use an array	441
Deleting sticky notes	446
The deleteSticky function	449
How do you select a sticky to delete?	450
How to get the sticky to delete from the event	451
Delete the sticky from the DOM, too	452
Update the user interface so we can specify a color	453
JSON.stringify, it's not just for Arrays	454
Using the new stickyObj	455
Don't Try This at Home (or Blowing Up Your 5 Megabytes)	458
Now that you know localStorage, how are you going to use it?	462
Bullet Points	464
Exercise Solutions	466

It's hard to manage my busy life if I can't get rid of these stickies after I'm done with them. Can you add a delete function?

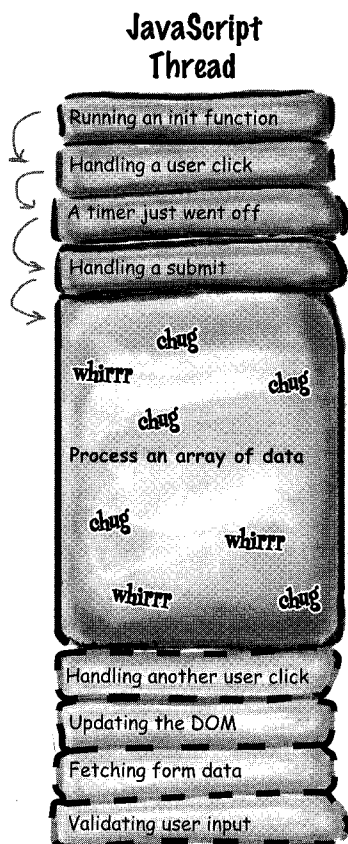


putting javascript to work

Web Workers

10

Slow script—do you want to continue running it? If you've spent enough time with JavaScript or browsing the web you've probably seen the "slow script" message. And, with all those multicore processors sitting in your new machine how could a script be running *too slow*? It's because JavaScript can only do one thing at a time. But, with HTML5 and Web Workers, *all that changes*. You've now got the ability to spawn *your own* JavaScript workers to get more work done. Whether you're just trying to design a more responsive app, or you just want to max out your machine's CPU, Web Workers are here to help. Put your JavaScript manager's hat on, let's get some workers cracking!

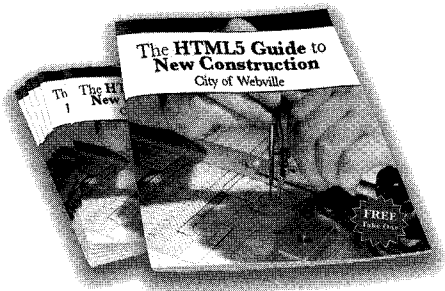


The Dreaded Slow Script	474
How JavaScript spends its time	474
When single-threaded goes BAD	475
Adding another thread of control to help	476
How Web Workers work	478
Your first Web Worker...	483
Writing Manager.js	484
Receiving messages from the worker	485
Now let's write the worker	486
Virtual Land Grab	494
How to compute a Mandelbrot Set	496
How to use multiple workers	497
Let's build the Fractal Explorer app	503
Ready Bake Code	504
Creating workers, and giving them tasks...	508
Writing the code...	509
Getting the workers started	510
Implementing the worker	511
Back to the code: how to process the worker's results	514
Fitting the canvas to the browser window	517
The anal-retentive chef coder	518
In the Laboratory	520
Bullet Points	524
Exercise Solutions	526

appendix: leftovers

We covered a lot of ground, and you're almost finished with this book.

We'll miss you, but before we let you go, we wouldn't feel right about sending you out into the world without a little more preparation. We can't possibly fit everything you'll need to know into this relatively small chapter. Actually, we *did* originally include everything you need to know about HTML5 (not already covered by the other chapters), by reducing the type point size to .00004. It all fit, but nobody could read it. So, we threw most of it away, and kept the best bits for this Top Ten appendix.



#1 Modernizr	532
#2 Audio	533
#3 jQuery	534
#4 XHTML is dead, long live XHTML	536
#5 SVG	537
#6 Offline web apps	538
#7 Web sockets	539
#8 More canvas API	540
#9 Selectors API	542
#10 But, there's even more!	543
The HTML5 Guide to New Construction	545
Webville Guide to HTML5 Semantic Elements	546
Webville Guide to CSS3 Properties	548