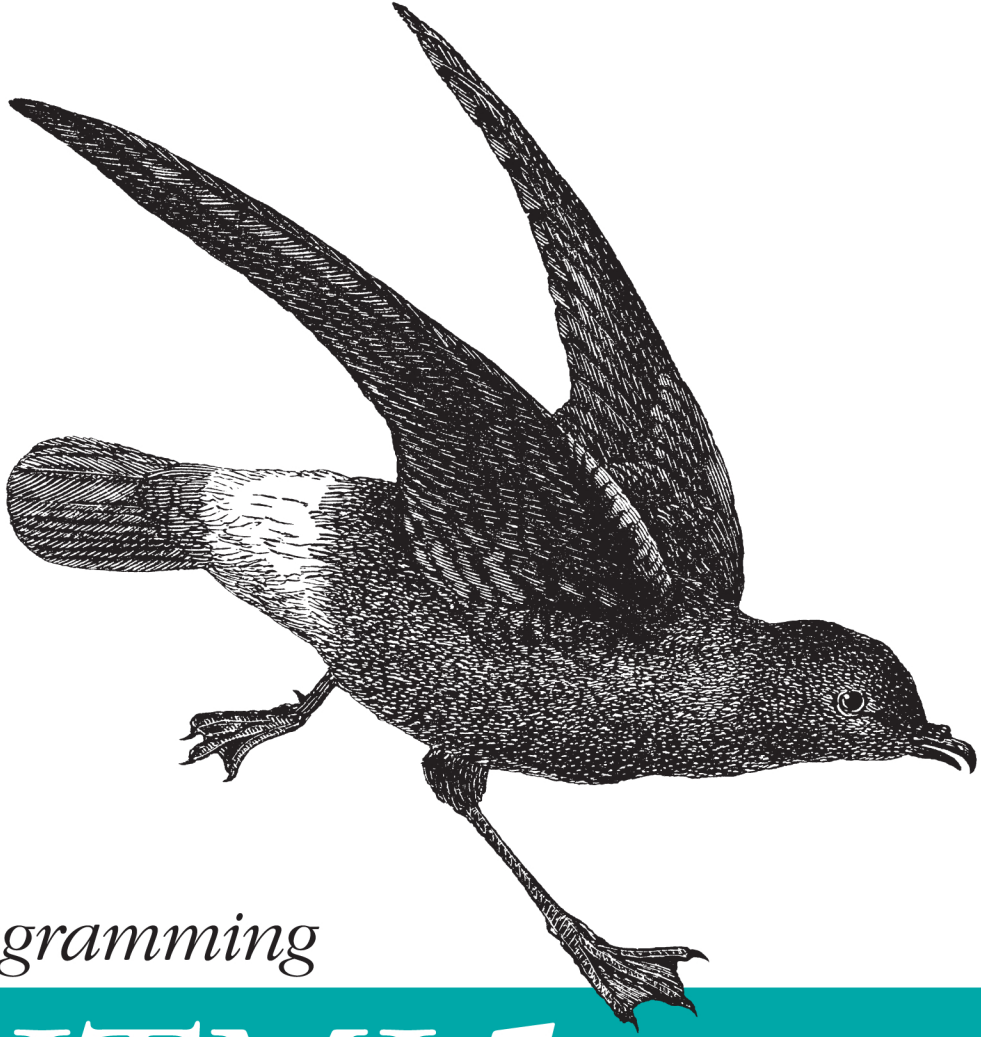


*Building Powerful Cross-Platform Environments in JavaScript*



*Programming*

# HTML5 Applications

O'REILLY®

*Zachary Kessin*

# Programming HTML5 Applications

Tap the full power of HTML5 and learn how to build complete, self-contained applications that can run on mobile devices and compete with desktop apps. With this practical guide, you'll discover effective HTML5 methods for working with data, such as offline storage and multithreaded processing. You'll also become familiar with advanced JavaScript tools that help you exploit the entire array of HTML5 elements. Packed with code samples, this book shows experienced JavaScript developers how HTML5 turns the Web into a first-class development environment.

- Store session data in the browser with local storage objects
- Save trips to the server: store larger amounts of data with IndexedDB
- Give browsers limited access to a user's system to read and upload files
- Take your app offline—and speed up page loading when it's online
- Use Web Workers to create multithreaded applications
- Transfer data between browser and server more efficiently with Web Sockets
- Learn about HTML5 tags for forms, multimedia, graphics, and geolocation

Zachary Kessin has been developing interactive web applications for two decades. He focuses on building complex applications in the browser with Javascript, browser-based testing with Selenium, functional programming, and code generation.

*“HTML5 is all the rage these days, but it's more than just a buzzword. Programming HTML5 Applications provides the knowledge to guide you through all the new technologies needed to make modern web applications.”*

—Ralph Whitbeck  
Cohost of The Official  
jQuery Podcast

US \$24.99

CAN \$26.99

ISBN: 978-1-449-39908-5



Twitter: @oreillymedia  
facebook.com/oreilly

**O'REILLY®**  
oreilly.com

---

# Programming HTML5 Applications

*Zachary Kessin*

**O'REILLY®**

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

# Programming HTML5 Applications

by Zachary Kessin

Copyright © 2012 Zachary Kessin. All rights reserved.

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (<http://my.safaribooksonline.com>). For more information, contact our corporate/institutional sales department: (800) 998-9938 or [corporate@oreilly.com](mailto:corporate@oreilly.com).

**Editors:** Andy Oram and Simon St. Laurent

**Production Editor:** Jasmine Perez

**Copyeditor:** Audrey Doyle

**Proofreader:** Kiel Van Horn

**Indexer:** Jay Marchand

**Cover Designer:** Karen Montgomery

**Interior Designer:** David Futato

**Illustrator:** Robert Romano

November 2011: First Edition.

## Revision History for the First Edition:

2011-11-8 First release

See <http://oreilly.com/catalog/errata.csp?isbn=9781449399085> for release details.

Nutshell Handbook, the Nutshell Handbook logo, and the O'Reilly logo are registered trademarks of O'Reilly Media, Inc. *Programming HTML5 Applications*, the image of a European storm petrel, and related trade dress are trademarks of O'Reilly Media, Inc.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly Media, Inc., was aware of a trademark claim, the designations have been printed in caps or initial caps.

While every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

ISBN: 978-1-449-39908-5

[LSI]

1320769400

---

# Table of Contents

<b>Preface .....</b>	<b>vii</b>
 <b>1. The Web As Application Platform .....</b>	 <b>1</b>
Adding Power to Web Applications	1
Developing Web Applications	2
JavaScript's Triumph	4
 <b>2. The Power of JavaScript .....</b>	 <b>7</b>
Nonblocking I/O and Callbacks	7
Lambda Functions Are Powerful	9
Closures	11
Functional Programming	13
Prototypes and How to Expand Objects	16
Expanding Functions with Prototypes	18
Currying and Object Parameters	21
Array Iteration Operations	22
You Can Extend Objects, Too	25
 <b>3. Testing JavaScript Applications .....</b>	 <b>27</b>
QUnit	30
A Simple Example	30
Testing with QUnit	32
Selenium	33
Selenium Commands	35
Constructing Tests with the Selenium IDE	38
Automatically Running Tests	39
Selenese Command Programming Interface	42
Running QUnit from Selenium	45
Selenium RC and a Test Farm	46

<b>4. Local Storage .....</b>	<b>49</b>
The localStorage and sessionStorage Objects	50
Using localStorage in ExtJS	53
Offline Loading with a Data Store	55
Storing Changes for a Later Server Sync	57
jQuery Plug-ins	58
DSt	58
jStore	59
<b>5. IndexedDB .....</b>	<b>61</b>
Adding and Updating Records	64
Adding Indexes	65
Retrieving Data	65
Deleting Data	66
<b>6. Files .....</b>	<b>67</b>
Blobs	67
Working with Files	69
Uploading Files	70
Drag-and-Drop	71
Putting It All Together	71
Filesystem	73
<b>7. Taking It Offline .....</b>	<b>75</b>
Introduction to the Manifest File	75
Structure of the Manifest File	76
Updates to the Manifest File	77
Events	79
Debugging Manifest Files	81
<b>8. Splitting Up Work Through Web Workers .....</b>	<b>85</b>
Web Worker Use Cases	87
Graphics	87
Maps	88
Using Web Workers	88
The Worker Environment	88
Worker Communication	89
Web Worker Fractal Example	90
Testing and Debugging Web Workers	96
A Pattern for Reuse of Multithread Processing	97
Libraries for Web Workers	101

<b>9. Web Sockets .....</b>	<b>103</b>
The Web Sockets Interface .....	105
Setting Up a Web Socket .....	105
Web Socket Example .....	106
Web Socket Protocol .....	108
Ruby Event Machine .....	108
Erlang Yaws .....	109
<b>10. New Tags .....</b>	<b>111</b>
Tags for Applications .....	111
Accessibility Through WAI-ARIA .....	112
Microdata .....	113
New Form Types .....	114
Audio and Video .....	115
Canvas and SVG .....	115
Geolocation .....	116
New CSS .....	116
<b>Appendix: JavaScript Tools You Should Know .....</b>	<b>117</b>
<b>Index .....</b>	<b>121</b>





---

# Preface

This book reflects the evolution of the Web. Less and less can programming be treated as a distinct activity shoehorned into web pages through scripts. Instead, HTML and JavaScript are now intertwined in producing an enchanting user experience. With this book, you can master the latest in this evolution.

## How This Book Is Organized

The elements of this book are as follows:

*Chapter 1, The Web As Application Platform*

Introduces the reasons for programming on the new HTML5 platforms and what they offer to the JavaScript programmer

*Chapter 2, The Power of JavaScript*

Explains some powerful features of JavaScript you may not already know, and why you need to use them to exploit the HTML5 features and associated libraries covered in this book

*Chapter 3, Testing JavaScript Applications*

Shows how to create and use tests in the unique environment provided by JavaScript and browsers

*Chapter 4, Local Storage*

Describes the `localStorage` and `sessionStorage` objects that permit simple data caching in the browser

*Chapter 5, IndexedDB*

Shows the more powerful NoSQL database that supports local storage

*Chapter 6, Files*

Describes how to read and upload files from the user's system

*Chapter 7, Taking It Offline*

Describes the steps you must go through to permit a user to use your application when the device is disconnected from the Internet

## Chapter 8, *Splitting Up Work Through Web Workers*

Shows the multithreading capabilities of HTML5 and JavaScript

## Chapter 9, *Web Sockets*

Shows how to transfer data between the browser and server more efficiently by using web sockets

## Chapter 10, *New Tags*

Summarizes tags introduced in HTML5 that are of particular interest to the web programmer

## Appendix, *JavaScript Tools You Should Know*

Describes tools used in the book, and others that can make coding easier and more accurate

# Conventions Used in This Book

The following typographical conventions are used in this book:

### *Italic*

Indicates new terms, URLs, email addresses, filenames, and file extensions

### **Constant width**

Used for program listings, as well as within paragraphs to refer to program elements such as variable or function names, databases, data types, environment variables, statements, and keywords

### **Constant width bold**

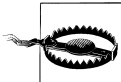
Shows commands or other text that should be typed literally by the user

### *Constant width italic*

Shows text that should be replaced with user-supplied values or by values determined by context



This icon signifies a tip, suggestion, or general note.



This icon indicates a warning or caution.

## Using Code Examples

This book is here to help you get your job done. In general, you may use the code in this book in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing a CD-ROM of examples from O'Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation does require permission.

We appreciate, but do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: "*Programming HTML5 Applications* by Zachary Kessin (O'Reilly). Copyright 2012 Zachary Kessin, 978-1-449-39908-5."

If you feel your use of code examples falls outside fair use or the permission given here, feel free to contact us at [permissions@oreilly.com](mailto:permissions@oreilly.com).

## Safari® Books Online



Safari Books Online is an on-demand digital library that lets you easily search more than 7,500 technology and creative reference books and videos to find the answers you need quickly.

With a subscription, you can read any page and watch any video from our library online. Read books on your cell phone and mobile devices. Access new titles before they are available for print, and get exclusive access to manuscripts in development and post feedback for the authors. Copy and paste code samples, organize your favorites, download chapters, bookmark key sections, create notes, print out pages, and benefit from tons of other time-saving features.

O'Reilly Media has uploaded this book to the Safari Books Online service. To have full digital access to this book and others on similar topics from O'Reilly and other publishers, sign up for free at <http://my.safaribooksonline.com>.

## How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.  
1005 Gravenstein Highway North  
Sebastopol, CA 95472  
800-998-9938 (in the United States or Canada)  
707-829-0515 (international or local)  
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at:

<http://shop.oreilly.com/product/0636920015116.do>

To comment or ask technical questions about this book, send email to:

[bookquestions@oreilly.com](mailto:bookquestions@oreilly.com)

For more information about our books, courses, conferences, and news, see our website at <http://www.oreilly.com>.

Find us on Facebook: <http://facebook.com/oreilly>

Follow us on Twitter: <http://twitter.com/oreillymedia>

Watch us on YouTube: <http://www.youtube.com/oreillymedia>

## Acknowledgments

A book is a team effort, and I could not have written this book without a great team behind me. First of all, I must thank Simon St. Laurent for giving me the chance to write this book and supporting me through the process of putting it together. I must also thank Andy Oram for his editorial prowess and ability to make the book better. Also, thank you to my technical reviewers, Shelley Powers and Dionysios Synodinos, for great feedback.

I must also thank the Israeli developer community for existing: my former coworkers at Mytopia, who supported me in this project for more than a year, and the gang at Sayeret Lambda, which has become the place in Tel Aviv to talk about programming.

Finally, I would like to thank my wife, Devora, for all her support in this project. I could not have done it without you.

# The Web As Application Platform

HTML5 makes the Web a first-class environment for creating real applications. It reinforces JavaScript's existing tool set with key extensions to the browser APIs that make it easier to create applications that feel (and can be) complete in themselves, not just views on some distant server process.

The Web began as a way to share files, stored on a web server, that changed only occasionally. Developers quickly figured out how to generate those files on the fly, taking the first big step toward building applications. The next big step was adding interactivity in the browser client. JavaScript and the Document Object Model (DOM) let developers create Dynamic HTML, as the “browser wars” raged and then suddenly stopped. After a few years, Ajax brought these techniques back into style, adding some tools to let pages communicate with the server in smaller chunks.

HTML5 builds on these 20 years of development, and fills in some critical gaps. On the surface, many of HTML5's changes add support for features (especially multimedia and graphics) that had previously required plug-ins, but underneath, it gives JavaScript programmers the tools they need to create standalone (or at least more loosely tethered) applications using HTML for structure, CSS for presentation, and JavaScript for logic and behavior.

## Adding Power to Web Applications

HTML5 raises the bar for web applications. While it still has to work under security constraints, it finally provides tools that desktop developers have expected for years:

### *Local data storage*

It can store up to 5 MB of data, referenced with a key-value system.

### *Databases*

Originally a SQLite-based API, the tide seems to have shifted to IndexedDB, a NoSQL system that is natively JavaScript.

### *Files*

While applications still can't freely access the filesystem (for obvious security reasons), they can now work with files the user specifies and are starting to be able to create files as well.

### *Taking it offline*

When a laptop or phone is in airplane mode, web applications are not able to communicate with the server. Manifest files help developers work around that by caching files for later use.

### *Web Workers*

Threads and forks have always been problematic, but JavaScript simply didn't offer them. Web Workers provide a way to put application processes into separate spaces where they can work without blocking other code.

### *Web sockets*

Hypertext Transfer Protocol (HTTP) has been the foundation of the Web, despite a few updates over time. Web sockets transform the request-response approach to create much more flexible communication systems.

There's much more, of course—from geolocation to audio and video to Canvas graphics to a wide variety of minor new tags—but these provide the foundations for building industrial-strength applications in HTML5.

## **Developing Web Applications**

In the old days, a complex web application might be a catalog, which would be static pages derived from a database, or a JavaScript loan calculator. But no one would have dreamed of doing complex applications in JavaScript. Those required Java or maybe a dedicated client/server application written in C or C++. Indeed, in the days before the DOM and Ajax, developing complex applications in JavaScript would have been pretty much impossible. However, Ajax introduced the ability to interact with the server without reloading the page, and the DOM allowed the programmer to change HTML on the fly.

In 2007, Google introduced Gears, a browser extension that gave the developer a lot more power than had been there before. Gears allowed the browser to work offline, to enable users to store more data in the browser and have a worker pool to offload long-running tasks. Gears has since been discontinued, as most of its features have migrated into HTML5 in modified forms.

The modern Web features a full range of sites, from things that are still effectively old-style collections of documents, like Wikipedia, to sites that offer interactions with other people, such as Facebook, YouTube, and eBay, to things that can serve as replacements for desktop applications, such as Gmail and Google Docs. Many formerly standalone applications, such as mail clients, have become part and parcel of the web experience.

In the modern Web, the line between applications and pages has blurred. The difference at this point is only in the intent of the site.

Running an application in the browser has some major advantages for both the user and the developer. For the user, there is no commitment to the application: you try it out, and if you don't like it, you can move on to the next page with nothing left behind to clutter up your disk. Trying new applications is also reasonably safe, in that they run in a sandboxed environment. New versions of the application are automatically downloaded to the browser when the developer updates the code. Web applications rarely have version numbers, at least public ones.

For the developer, the case is even stronger. First of all, the things that are an advantage to the users are also good for the developers. There is no installation program to write, and new versions can automatically be sent to the users, making small, incremental updates not only possible but practical. However, there are other bonuses as well.

The Web is cross-platform. It is possible to write a web page that will work on Windows XP, Windows Vista, Windows 7, Mac OS X, Linux, the iPhone/iPad, and Android. Doing that with a conventional development tool would be a monumental task. But with the Web and some forethought it almost comes for free. A web application built on standards with a library like jQuery will be able to run on major browsers on all those platforms and a few others. While at one point Sun hoped that its Java applets would define the Web as a platform, JavaScript has turned out to become the default web platform.

You can even run web applications on mobile devices, at least the ones that today are called smartphones. With a wrapper like PhoneGap, you can create an HTML5 app and package it for sale in the App Store, the Android Market, and more. You might create an application that interacts heavily with a web server, or you might create a completely self-contained application. Both options are available.

The real place that the Web, prior to HTML5, traditionally falls short is that a web application, running on a computer with gigabytes of memory and disk space, acts almost like it is running on an old VT320 terminal. All data storage must be done on a server, all files must be loaded from the server, and every interaction pretty much requires a round-trip to the server. This can cause the user experience to feel slow, especially if the server is far away from the user. If every time the user wishes to look up something there is a minimum response time of 400 milliseconds before any actions can be taken, the application will feel slow. From my office in Tel Aviv to a server in California, the round-trip time for an ICMP ping is about 250 ms. Any action on the server would be extra and slow that down even more. Mobile device communications can, of course, be even slower.