O'REILLY®

Enterprise Web Development

企业级Web开发（影印版）

Yakov Fain, Victor Rasputnis,
Anatole Tartakovsky, Viktor Gamov 著

東南大學出版社

企业级Web开发（影印版）

Enterprise Web Development

Yakov Fain, Victor Rasputnis,
Anatole Tartakovsky, Viktor Gamov 著

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo O'REILLY®

O'Reilly Media, Inc.授权东南大学出版社出版

南京 东南大学出版社

图书在版编目(CIP)数据

企业级 Web 开发:英文/(美)费恩(Fain, Y.)等著.
影印本. —南京:东南大学出版社,2015.8
书名原文:Enterprise Web Development
ISBN 978-7-5641-5916-0

Ⅰ.①企… Ⅱ.①费… Ⅲ.①网页制作工具—
程序设计—英文 Ⅳ.①TP393.092

中国版本图书馆 CIP 数据核字(2015)第 165412 号

江苏省版权局著作权合同登记
图字:10-2015-158 号

企业级 Web 开发(影印版)

出版发行:东南大学出版社
地　　址:南京四牌楼 2 号　　邮编:210096
出 版 人:江建中
网　　址:http://www.seupress.com
电子邮件:press@seupress.com
印　　刷:常州市武进第三印刷有限公司
开　　本:787 毫米×980 毫米　　16 开本
印　　张:40.25
字　　数:788 千字
版　　次:2015 年 8 月第 1 版
印　　次:2015 年 8 月第 1 次印刷
书　　号:ISBN 978-7-5641-5916-0
定　　价:98.00 元

本社图书若有印装质量问题,请直接与营销部联系。电话(传真):025-83791830

Preface

This book will help web application developers and software architects pick the right strategy for developing cross-platform applications that run on a variety of desktop computers as well as mobile devices. The primary audience is developers from a large organization who need to learn how to develop web applications using the HTML5 stack.

What's an Enterprise Application?

This book has the word *enterprise* in its title, and we'll explain what we consider to be *enterprise applications* by giving you some examples. Creating a web application that processes orders is not the same as creating a website to publish blogs. Enterprise applications require company-specific workflows, which usually need to be integrated with various internal systems, data sources, and processes.

Google Docs is not an enterprise web application. But Google Search Appliance, which integrates search operations with company documents, databases, processes, and tickets, and provides collaboration is: it integrates the consumer-workforce front office with what the company does (back office).

Google Maps is not an enterprise application. But Google Maps integrated with a company site used by insurance agents to plan their daily route, create scheduling, perform address verification, and use geocoding is.

Just using a web application in a business doesn't make it an enterprise web application. If you take Gmail as is, it won't be an enterprise application until you integrate it into another process of your business.

Is an online game an enterprise application? It depends on the game. A multiplayer online roulette game hooked up to a payment system and maintaining users' accounts is an enterprise web application. But playing Sudoku online doesn't feel too enterprisey.

How about a dating website? If the site just offers an ability to display singles, it's just a publishing site because there is not much of a business there. Can you turn a dating website into an enterprise application? It's possible.

Some people will argue that an enterprise application must support multiple users and a high data load, include data grids and dashboards, be scalable, have business and persistence layers, offer professional support, and more. This is correct, but we don't believe that a web application should do all this to qualify for the adjective *enterprise*.

Let's create a simple definition of an enterprise web application:

An enterprise web application is one that helps an organization run its business online.

Why the Authors Wrote This Book

The authors of this book have 90 years of combined experience in developing enterprise applications. During all these years, we've been facing the same challenges regardless of which programming language we use:

- How to make the application code base maintainable
- How to make the application responsive by modularizing its code base
- How to minimize the number of production issues by applying proper testing at earlier stages of the project life cycle
- How to design a UI that looks good and is convenient for users
- Which frameworks or libraries to pick
- Which design patterns to apply in coding

This list can be easily extended. Ten years ago, we were developing UIs mainly in Java; five years ago, we used Adobe Flex; today, we use HTML5-related technologies. This book shares with you our understanding of how to approach these challenges in HTML5.

Who This Book Is For

Web application development with HTML5 includes HTML, JavaScript, CSS, and dozens of JavaScript frameworks. The main goal of this book is to give you a hands-on overview of developing web applications that can be run on a variety of devices—desktops, tablets, and smartphones. We expect you to have some experience with any programming language. Knowledge of basic HTML is also required. Understanding the principles of object-oriented programming would be helpful, too.

This book is intended for software developers, team leaders, and web application architects who want to learn the following:

- How to write web applications by using some of the popular libraries and frameworks
- How to modularize the client's side of web applications written in JavaScript
- How to test web applications
- Whether applying *responsive design* principles is the right strategy for your application
- Which security vulnerabilities to watch for
- Why developing for mobile devices differs from developing for desktops
- The pros and cons of developing mobile applications by using the HTML5 stack versus native languages

If you're new to programming in JavaScript, start reading this book from the bonus online chapter (*https://github.com/oreillymedia/enterprise_web_development*), which is an introduction to JavaScript.

What This Book Is and Why It's Important

This book has a lot of breadth, but for mastering some of the topics in depth, be prepared to do additional studying. On the other hand, we provide a lot of working code samples for those who prefer studying by reading code.

This book can be important for busy professionals who don't have time to read a separate book about each and every library and framework that exist in the HTML5 universe. This book will help you to narrow the list of technologies and frameworks to be considered for the next project.

Enterprise server-side developers will also benefit from reading this book. Pretty often enterprise Java or .NET developers feel caught off guard when they need to create a new web application with a cross-platform and cross-browser UI. These strong enterprise developers with good business knowledge may not have enough exposure to how things work in the HTML5 domain. This book can be a time-saver for all server-side developers who need to start working on the frontend of web applications.

Finally, this book is important because of the way it's written: you'll be working on the application that's introduced next.

Introducing the Save The Child Application

To make this book more practical, we decided not to give you unrelated code snippets illustrating various syntax or techniques, but to bring all of that together in a working application (just the UI portion). While learning the various frameworks, libraries, and approaches to building UIs for web applications, you'll be writing multiple versions of the same web application—Save The Child (see Figure P-1). It's a sample charity application used to collect donations for children who need medical attention.

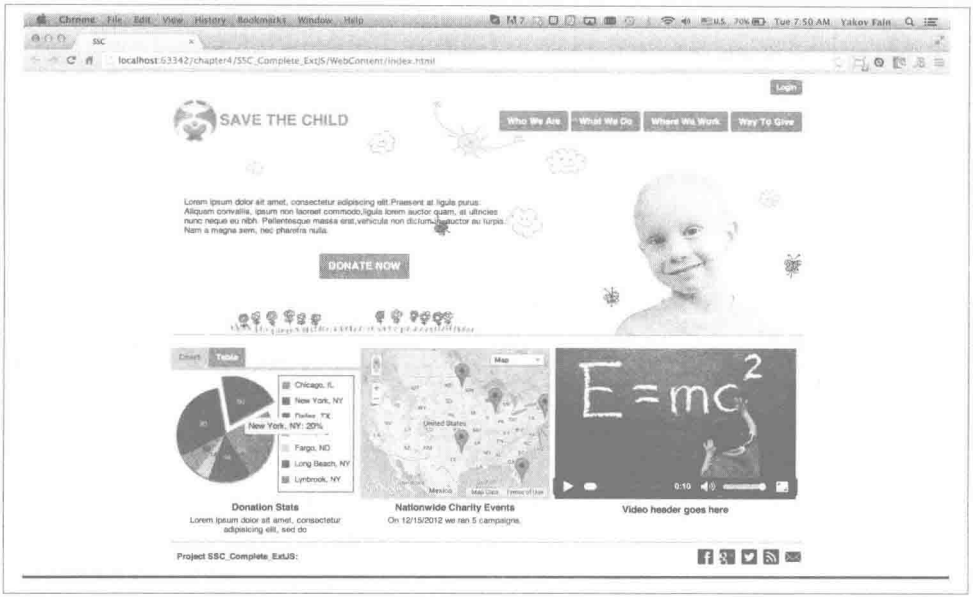

Figure P-1. Save The Child—a sample application

This web application will allow people to register, donate, find local kids who need help, match donors and recipients, upload images and videos, and display statistics.

Is This Even an Enterprise App?

While looking at the preceding image, you might be thinking, "This doesn't look like an enterprise application." Let's see. Do you believe that an enterprise application has to consist of boring gray windows with lots of grids and forms, and some charts? True, but we have all of these elements in our application, too:

- Clicking the Donate Now button reveals a form that has to be filled out and sent to a payment processing system.

- The interactive live pie chart is something that many modern enterprise dashboards include.
- Clicking the Table tab (right next to the Chart tab) shows the same donation stats in a grid (that one is grayish).
- Integration with the mapping API allows you to visually present the locations of important events for this business or nonprofit organization.
- Under the hood, this pretty window will use the high-speed, full-duplex communication protocol WebSocket.

As a matter of fact, the company that employs the authors of this book has a customer that is a nonprofit organization that is in the business of helping people fighting a certain disease. That application has two parts: consumer-facing and back-office. The former looks more colorful, whereas the latter has more gray grids indeed. Both parts process the same data, and this organization can't operate if you remove either of these parts.

Would these features make Save The Child an enterprise web application? Yes, because it can help our imaginary nonprofit organization run its business: collecting donations for sick kids. Would you rather see a fully functioning Wall Street trading system? Maybe. But this book and our sample application incorporate all software components that you'd need to use for developing a financial application.

How We Are Going to Build This App

Instead of presenting unrelated code samples, we decided to develop multiple versions of the same web application, built with different libraries, frameworks, and techniques. This approach allows you to compare apples to apples and to make an educated decision about which approach best fits your needs.

First, we'll show how to build this application in pure HTML/JavaScript. Then, we'll rewrite it using the jQuery library, and then with the Ext JS framework. Users will be able to see where different charity events are being run (via Google Maps integration). The page will integrate a video player and display a chart with stats on donors by geographical location. One of the versions shows how to modularize the application; this is a must for any enterprise system. Another version shows how to use WebSocket technology to illustrate the server-side data push while adding an auction to this web application. The final chapters of the book show various ways of building different versions of the same Save The Child application to run on mobile devices (responsive design, jQuery Mobile, Sencha Touch, and PhoneGap). We believe that this application will help you to compare all these approaches and select those that fit your objectives.

The Goals of the Book

First, we want to say what's not the goal of this book: we are not planning to convince you that developing a cross-platform web application is the right strategy for you. Don't be surprised if, after reading this book, you decide that developing applications in HTML5 is not the right approach for the tasks you have at hand. This book should help decision makers pick the right strategy for developing cross-platform applications that run on a variety of desktop computers as well as mobile devices.

Technologies Used in This Book

This is an HTML5 book, and the main programming language used here is JavaScript. We use HTML and CSS, too. Most JavaScript development is done using various libraries and frameworks. The difference between a *library* and a *framework* is that the former does not dictate how to structure the code of your application; a library simply offers a set of components that will spare you from writing lots of manual code. The goal of some frameworks is to help developers test their applications. The goal of other frameworks is just to split the application into separate modules. There are tools just for building, packaging, and running JavaScript applications. Although many of the frameworks and tools are mentioned in this book, the main technologies/libraries/tools/techniques/protocols used in this book are listed here:

- Balsamiq Mockups
- Modernizr
- jQuery
- jQuery Mobile
- Ext JS
- Sencha Touch
- RequireJS
- Jasmine
- Clear Data Builder
- WebSocket
- PhoneGap
- Grunt
- Bower
- WebStorm IDE
- Eclipse IDE

Although you can write your programs in any text editor, using specialized integrated development environments (IDEs) is more productive, and we'll use the Aptana Studio IDE by Appcelerator and the WebStorm IDE by JetBrains.

How the Book Is Organized

Even though you may decide not to read some of the chapters, we still recommend that you to skim through them. If you're not familiar with JavaScript, start from the online bonus chapter (*http://bit.ly/1iJO41S*).

Chapters 1 and 2 are must reads; if you can't read JavaScript code or are not familiar with CSS, Ajax, or JSON, the rest of the book will be difficult to understand. On the other hand, if you're not planning to use, say, the Ext JS framework, you can just skim through Chapter 4. Following is a brief book outline.

The Preface includes a brief discussion of the difference between enterprise web applications and websites. It also touches on the evolution of HTML.

Chapter 1 describes the process of mocking up the application Save The Child, which will solicit donations to children, embed a video player, integrate with Google Maps, and eventually feature an online auction. We show you how to gradually build all the functionality of this web application while explaining each step of the way. By the end of this chapter, we'll have the web design and the first prototype of the Save The Child application written using just HTML, JavaScript, and CSS.

Chapter 2 is about bringing external data to web browsers by making asynchronous calls to a server. The code from the previous chapter uses only hardcoded data. Now it's time to learn how to make asynchronous server calls by using Ajax techniques and consume the data in JSON format. The Save The Child application will start requesting the data from the external sources and sending them the JSON-formatted data.

Chapter 3 shows how to use a popular jQuery library to lower the amount of manual coding in the Save The Child application. First, we introduce the jQuery Core library, and then rebuild our Save The Child application with it. In the real world, developers often increase their productivity by using JavaScript libraries and frameworks.

Chapter 4 is a mini tutorial of a comprehensive JavaScript framework called Ext JS. This is one of the most feature-complete frameworks available on the market. Sencha, the company behind Ext JS, has managed to extend JavaScript to make its syntax closer to classical object-oriented languages. Sencha also developed an extensive library of the UI components. Expect to see another rewrite of the Save The Child application here.

Chapter 5 is a review of productivity tools (including npm, Grunt, Bower,Yeoman, and CDB) used by enterprise developers. It's about using build tools, working with code generators, and managing dependencies (a typical enterprise application uses various software that needs to work in harmony).

Chapter 6 explains how to modularize large applications. Reducing startup latency and implementing lazy loading of certain parts of the application are the main reasons for modularization. We give you an example of how to build modularized web applications that won't bring large, monolithic code to the client's machine, but rather loads the code on an as-needed basis. You'll also see how to organize the data exchange between programming modules in a loosely coupled fashion. The Save The Child application is rewritten with the RequireJS framework, which will load modules on demand rather than the entire application.

Chapter 7 is dedicated to test-driven development with JavaScript. To shorten the development cycle of your web application, you need to start testing it in the early stages of the project. It seems obvious, but many enterprise IT organizations haven't adopted agile testing methodologies, which costs them dearly. JavaScript is dynamically typed interpreted language—there is no compiler to help identify errors as it's done in compiled languages like Java. This means that a lot more time should be allocated for testing JavaScript web applications. We cover the basics of testing and introduce you to some of the popular testing frameworks for JavaScript applications. Finally, you'll see how to test the Save The Child application with the Jasmine framework.

Chapter 8 shows how to substantially speed up interactions between the client and the server by using the WebSocket protocol introduced in HTML5. HTTP adds a lot of overhead for every request and response object that serve as wrappers for the data. You'll see how to introduce a WebSocket-based online auction to the new version of our Save The Child application. This is what Ian Hickson, the HTML5 spec editor from Google, said about why the WebSocket protocol is important:

> *Reducing kilobytes of data to 2 bytes is more than a little more byte efficient, and reducing latency from 150 ms (TCP round-trip to set up the connection plus a packet for the message) to 50 ms (just the packet for the message) is far more than marginal. In fact, these two factors alone are enough to make WebSocket seriously interesting to Google.*

Chapter 9 is a brief introduction to web application security. You'll learn about vulnerabilities of web applications and will get references to recommendations on how to protect your application from attackers. This chapter concludes with some of the application-specific security considerations (like regulatory compliance) that your business customers can't ignore.

Chapter 10 opens up a discussion of how to approach creating web applications that should run not only on desktops, but also on mobile devices. In this chapter, you become familiar with the principles of responsive design, which allow you to have a single code base that will be flexible enough to render a UI that looks good on large and small screens. You'll see the power of CSS *media queries* that automatically reallocate UI components based on screen width of the device on which the website is being viewed. The new version of the Save The Child application will demonstrate how to go about responsive design.

Chapter 11 introduces you to jQuery Mobile—a library that was specifically created for developing mobile web applications. But main principles implemented in the larger jQuery library remain in place, and studying the materials from Chapter 3 is a prerequisite for understanding this chapter. Then you'll create the mobile version of the Save The Child application with jQuery Mobile.

Chapter 12 is about a little brother of Ext JS—Sencha Touch. This framework was developed for mobile devices, and you'll need to read Chapter 6 in order to understand the materials from this one. As usual, we develop another variation of the mobile version of the Save The Child application with Sencha Touch.

Chapter 13 shows how you can create hybrid mobile applications, which are written with HTML/JavaScript/CSS but can use the native API of the mobile devices. Hybrids are packaged as native mobile applications and can be submitted to popular online app stores or marketplaces the same way as if they were written in the programming language native for the mobile platform in question. This chapter illustrates how to access the camera of a mobile device by using the PhoneGap framework.

The bonus online chapter (*https://github.com/oreillymedia/enterprise_web_development*) is an introduction to programming with JavaScript. In about 60 pages, we cover the main aspects of this language. No matter what framework you choose, a working knowledge of JavaScript is required.

Appendix A is a brief overview of selected APIs from the HTML5 specification. They are supported by all modern web browsers. We find these APIs important and useful for many web applications. The following APIs are reviewed:

- Web Messaging
- Web Workers
- Application Cache
- Local Storage
- Indexed Database
- History API

Appendix B is a brief discussion of the IDEs that are being used for HTML5 development in general and in this book in particular.

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 element signifies a tip or suggestion.

This element signifies a general note.

This element indicates a warning or caution.

The Source Code for the Examples

The source code for all versions of the Save The Child application are available for download from O'Reilly at the book's catalog page (*http://bit.ly/enterprise-web-development*). There is also a GitHub repository (*http://bit.ly/1uFXI5u*) where the authors keep the source code of the book examples.

The authors also maintain the website (*http://savesickchild.org*), where various versions of the sample Save The Child application are deployed so you can see them in action.

This book is here to help you get your job done. In general, if example code is offered with this book, you may use it 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: "*Enterprise Web Development* by Yakov Fain, Victor Rasputnis, Anatole Tartakovsky, and Viktor Gamov (O'Reilly). Copyright 2014 Yakov Fain, Victor Rasputnis, Anatole Tartakovsky, and Viktor Gamov, 978-1-449-35681-1."

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at *permissions@oreilly.com*.

Safari® Books Online

Safari Books Online is an on-demand digital library that delivers expert content in both book and video form from the world's leading authors in technology and business.

Technology professionals, software developers, web designers, and business and creative professionals use Safari Books Online as their primary resource for research, problem solving, learning, and certification training.

Safari Books Online offers a range of product mixes and pricing programs for organizations, government agencies, and individuals. Subscribers have access to thousands of books, training videos, and prepublication manuscripts in one fully searchable database from publishers such as O'Reilly Media, Prentice Hall Professional, Addison-Wesley Professional, Microsoft Press, Sams, Que, Peachpit Press, Focal Press, Cisco Press, John Wiley & Sons, Syngress, Morgan Kaufmann, IBM Redbooks, Packt, Adobe Press, FT Press, Apress, Manning, New Riders, McGraw-Hill, Jones & Bartlett, Course Technology, and dozens more. For more information about Safari Books Online, please visit us online.

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://bit.ly/enterprise-web-development*.

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

You see four names on this book cover. But this book is a product of more than four people. It's a product of our company, Farata Systems.

In particular, we'd like to thank Alex Maltsev, who plays the a role of Jerry-the-Designer from Chapter 1 onward. Alex created all the UI prototypes for the sample web application Save The Child that is designed, redesigned, developed, and redeveloped several times in this book. He also developed code samples for the book and all CSS files.

Our big thanks to Anton Moiseev, who developed the Ext JS and Sencha Touch versions of our sample application.

Our hats off to the creators of the Asciidoc text format (*http://asciidoc.org/*). The drafts of this book were prepared in this format, with the subsequent generation of PDF, EPUB, MOBI, and HTML documents.

Our sample application uses two images from the iStockPhoto (*http://www.istockphoto.com/*) collection: the smiling boy by the user *jessicaphoto* and the logo by the user *khalus*. Thank you, guys!

Finally, our thanks to the O'Reilly editors for being so patient while we were trying to hit lots of moving and evolving targets that together represent the universe known as HTML5.

Introduction

During the last decade, the authors of this book worked on many enterprise web applications using a variety of programming languages and frameworks: HTML, JavaScript, Java, and Flex, to name a few. Apache Flex and Java produce compiled code that runs in a well-known and predictable virtual machine (JVM and Flash Player, respectively).

This book is about developing software by using what's known as the HTML5 stack. But only the second chapter of this book offers you an overview of the selected HTML5 tags and APIs. The first chapter is an advanced introduction to JavaScript. The rest of the chapters are about designing, redesigning, developing, and redeveloping a sample website for Save The Child. You'll be learning whatever is required for building this web application on the go.

You'll be using dynamic HTML (DHTML), which is HTML5, JavaScript, and Cascading Style Sheets (CSS). We'll add to the mix the `XMLHttpRequest` object that lives in a web browser and communicates with the server without the need to refresh the entire web page (a.k.a. Ajax). JSON will be our data format of choice for data exchange between the web browser and the server.

Moving from DHTML to HTML5

DHTML stands for *Dynamic HTML*. Back in 1999, Microsoft introduced the `XMLHttpRequest` object to allow the web version of its mail client, Outlook, to update the browser's window without the need to refresh the entire web page. Several years later, it was substituted with a more popular acronym, AJAX (which stood for Asynchronous JavaScript and XML); today we refer to it simply as a name, "Ajax." The market share of Internet Explorer 5 was about 90 percent at the time, and in enterprises it was literally the only approved browser.

Many years passed by, and today's Internet ecosystems have changed quite a bit. Web browsers are a lot smarter, and the performance of JavaScript has improved substantially.

The browsers support multiple simultaneous connections per domain (as opposed to 2 five years ago), which gave a performance boost to all Ajax applications. At least one-third of all web requests are being made from smartphones or tablets. Apple started its war against all browser plug-ins; hence using embedded Java VM or Flash Player is not an option there. The growing need to support a huge variety of mobile devices gave another boost for the HTML5 stack, which is supported by all devices.

But choosing HTML5 as the least common denominator that works in various devices and browsers means lowering requirements for your enterprise project. The UI might not be pixel-perfect on any particular device, but it will be made somewhat simpler (compared to developing for one specific VM, device, or OS) and will have the ability to adapt to different screen sizes and densities. Instead of implementing features that are specific to a particular device, the functional specification will include requirements to test under several web browsers, in many screen sizes and resolutions. HTML5 developers spend a lot more time in the debugger than people who develop for a known VM. You'll have to be ready to solve problems such as a drop-down not showing any data in one browser while working fine in others. Can you imagine a situation when the click event is not always generated while working in Java, Flex, or Silverlight? Get ready for such surprises while testing your HTML5 application.

You'll save some time because there is no need to compile JavaScript, but you'll spend more time testing the running application during development and Quality Assurance (QA) phases. The final deliverable of an HTML5 project might have as low as half of the functionality compared to the same project developed for a VM. But you'll gain a little better web adaptability, easier implementation of full-text search, and the ability to create mashups (*http://en.wikipedia.org/wiki/Mashup_(web_application_hybrid)*). Integration with other technologies will also become easier with HTML/JavaScript. If all these advantages are important to your applications, choose HTML5.

JavaScript will enforce its language and tooling limitations on any serious and complex enterprise project. You can develop a number of fairly independent windows, but creating well-tested and reliable HTML5 applications takes time. It can be significantly easier with the use of libraries or a framework.

In this book, we use some JavaScript frameworks; there are dozens on the market. Several of them promise to cover all the needs of your web application. Overall, there are two main categories of frameworks:

- Those that allow you to take an existing HTML5 website and easily add new attributes to all or some page elements so they would start shining, blinking, or do some other fun stuff. Such frameworks don't promote component-based development. They may not include navigation components, grids, or trees, which are pretty typical for any UI of the corporate tasks. JQuery is probably the best representative of this group; it's light (30 Kb), extendable, and easy to learn.