



Community Experience Distilled

Web Development with MongoDB and Node.js

Build an interactive and full-featured web application from scratch using Node.js and MongoDB

Jason Krol

[PACKT] open source*
PUBLISHING community experience distilled

Web Development with MongoDB and Node.js

Build an interactive and full-featured web application from scratch using Node.js and MongoDB

Jason Krol



BIRMINGHAM - MUMBAI

Web Development with MongoDB and Node.js

Copyright © 2014 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the authors, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: September 2014

Production reference: 1180914

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-78398-730-6

www.packtpub.com

Cover image by Jarek Blaminsky (milak6@wp.pl)

Credits

Author

Jason Krol

Reviewers

Anthony Gilardi

James O'Brien

Mithun Satheesh

Peter Shannon

Acquisition Editor

Llewellyn Rozario

Content Development Editor

Susmita Panda Sabat

Technical Editor

Faisal Siddiqui

Copy Editors

Mradula Hegde

Adithi Shetty

Project Coordinator

Neha Thakur

Proofreaders

Simran Bhogal

Maria Gould

Ameesha Green

Paul Hindle

Lucy Rowland

Indexers

Monica Ajmera Mehta

Priya Sane

Graphics

Abhinash Sahu

Production Coordinator

Nitesh Thakur

Cover Work

Nitesh Thakur

About the Author

Jason Krol is a passionate web developer with over 15 years of professional experience in creating highly interactive web applications using the latest in both client and server technologies.

Previously, Jason spent a majority of his career working with the Microsoft stack using ASP.net. Recently, he has been focusing on developing Single Page Applications using JavaScript in the full stack with Node.js, MongoDB, and Backbone.js. After co-owning and running a successful web development agency for a number of years, Jason recently jumped back into the world of full time employment.

When not writing code for work or side projects, he blogs about his development experiences and opinions at www.KrolTech.com and on Twitter at @ShortTompkins. He loves spending his free time with his wife and 8-year-old son.

I would like to specially thank my wonderful wife for putting up with me and for always being there to push me whenever I doubt myself.

About the Reviewers

Anthony Gilardi is a full stack JavaScript developer at one of the top e-mail marketing companies and a mobile technology enthusiast and app developer. He started programming at the age of 12 on a Sinclair ZX80. During his formative years, he worked on fighter jets and jet engines in the United States Air Force. He later earned a Bachelor of Science degree from Rutgers University in Biochemical Engineering. Programming, however, was his true passion, so he worked for over a decade doing Microsoft development in the pharmaceutical industry. He is now fully focused on working with JavaScript technologies for web and mobile development in what he considers the most exciting time to be a JavaScript developer.

Anthony's personal passion and creative energy is focused toward his *appMite* brand creating mobile hybrid apps. The introduction of Palm webOS and the Mojo framework ignited his interest in using JavaScript for mobile apps. He has created several hybrid apps, most notably his lifestyle application named lifeMite, which was released for Android, iOS, Kindle, and Nook. He is excited about the future of hybrid apps and learning budding technologies such as Polymer and Web Components.

If he doesn't know it, he learns it. If he doesn't understand it, he learns it more. Anthony loves creative whims. This is something he has always had but has been reinforced by his wife and three children who are always creating and learning. He doesn't ever want to lose that creative drive.

Outside of technology, he is a husband and father of three, who loves camping, walking, and finding obscure places to meditate. If you wish to see his latest apps or programming projects, visit <http://appmite.com>, or if you wish to read his personal ventures, please visit <http://journeysimple.com>.

James O'Brien is a software engineer with over 15 years of experience as a web technologist, specializing in professional web application development with technologies, including HTML5, JavaScript, CSS, Backbone.js, C#, ASP.NET, MVC, PHP, and SQL Server. He also has experience in graphic design and online marketing.

James is a Philadelphia native and is currently Manager for Web Development and Interactive Marketing at NextGen Healthcare. Besides this, he co-created Fill The Part (fillthepart.com) and runs it. Launched in 2012, it gives you a unique entertainment website experience: the chance to cast (or recast) a movie the way you'd want to see it. His other interests include movies (of course), playing basketball, and gaming. He's a proud husband and father.

Mithun Satheesh is an open source enthusiast and a full stack web developer from India. Starting his career as a PHP developer, he has over 4 years of experience in web development both in frontend and backend programming.

He has written a couple of libraries on Node.js and published them on npm, which have got a considerable user base. One of these is called *node-rules*, a forward-chaining rule engine implementation written initially to handle transaction risks on bookmyshow.com, one of his former employers. He is a regular on programming sites such as Stack Overflow and loves contributing to the open source world.

Apart from programming, he is also interested in experimenting with various PaaS solutions. He has a number of applications listed in the developer spotlight of PaaS providers such as Red Hat's OpenShift.

You can follow him on Twitter at [@mithunsatheesh](https://twitter.com/mithunsatheesh).

I would like to thank my parents for allowing me to live the life that I wanted to live. I am thankful to all my teachers for whatever knowledge I have gained in my life.

Peter Shannon is a husband and father who moonlights as a software engineer and data scientist. Originally a chemist, he found his passion for software engineering in high performance computing and computational chemistry – which never really became his forte. Now, he spends most of his time writing data-driven tools, staring at graphs, and occasionally doing math. When relaxing, he enjoys watching Star Trek Deep Space Nine reruns and pondering about life's biggest questions.

I would like to thank the Flying Spaghetti Monster for the noodly strength to review this book.

www.PacktPub.com

Support files, eBooks, discount offers, and more

You might want to visit www.PacktPub.com for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.PacktPub.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at service@packtpub.com for more details.

At www.PacktPub.com, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.



<http://PacktLib.PacktPub.com>

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print and bookmark content
- On demand and accessible via web browser

Free access for Packt account holders

If you have an account with Packt at www.PacktPub.com, you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

Table of Contents

Preface	1
Chapter 1: Welcome to JavaScript in the Full Stack	7
Node.js changed JavaScript forever	8
Asynchronous callbacks	9
Node Package Manager	10
Networking and file IO	10
Not just on the web	10
Real-time web with Socket.io	11
The NoSQL movement	11
Node and MongoDB in the wild	12
What to expect from this book	13
Summary	14
Chapter 2: Getting Up and Running	15
Environment assumptions and requirements	15
Installing Node.js	16
Mac OS X installation instructions	16
Windows 7 or 8 installation instructions	17
Linux installation instructions	18
Confirming successful Node.js installation	19
Bookmarking the online documentation	20
Installing the MongoDB server	20
Mac OS X installation instructions	21
Windows 7 or 8 installation instructions	22
Linux installation instructions	24
Confirming successful MongoDB installation	25
Bookmarking the online documentation	26

Writing your first app	26
The code	26
Launch the sample app	30
Check the actual database	30
Summary	31
Chapter 3: Node and MongoDB Basics	33
A JavaScript Primer	33
Declaring variables	34
Declaring functions	35
Declaring objects	36
Functions are objects	37
Anonymous functions and callbacks	38
Arrays	40
Conditions and comparison operators	40
Flow	41
JSON	42
The basics of NodeJS	43
Event driven	43
Asynchronous	43
Require and modules	44
The NodeJS core	44
Installing modules using npm	45
The basics of MongoDB	46
The mongo shell	47
Inserting data	47
Querying	48
Updating data	49
Deleting data	50
Additional resources	50
Summary	51
Chapter 4: Writing an Express.js Server	53
What is Express.js?	53
Building a complete web application	54
Organizing the files	56
Server.js – where it all begins	57
Bootting up server.js	58
Configuration module	59
Handlebars view engine	60
Other template engines	61

Using and understanding middleware	62
Introducing Connect	62
Activating the configure module	65
Routers and controllers	66
Custom middleware	71
Migrating to Express v4.0.0	72
Using new middleware	72
server/configure.js	73
server/routes.js	76
Summary	76
Chapter 5: Dynamic HTML with Handlebars	77
Basic syntax for Handlebars	77
Views	78
Layouts	85
Partial views	87
Handlebars Helpers	89
Global helpers	89
View-specific helpers	90
Rendering the views	91
Summary	93
Chapter 6: Controllers and View Models	95
Controllers	95
View models	96
Updating the home controller	97
Updating the image controller	100
Displaying an image	100
Uploading an image	102
Helpers for reusable code	106
The sidebar module	106
The stats module	108
The images module	109
The comments module	110
Testing the sidebar implementation	111
Iterating on the UI	112
Summary	116
Chapter 7: Persisting Data with MongoDB	117
Using MongoDB with Node	118
Connecting to MongoDB	119
Inserting a document	120
Retrieving a document	121

Introducing Mongoose	122
Schemas	123
Models	124
Built-in validation	126
Static methods	128
Virtual properties	128
Connecting with Mongoose	129
Defining the schema and models	130
Models index file	132
Adding CRUD to the controllers	133
The home controller	134
The image controller	136
Index – retrieving an image model	137
Create – inserting an image model	141
Like – updating an image model	146
Comment – inserting a comment model	148
Wrapping it up	150
Helpers	150
Introducing the async module	151
The comments helper	151
The helper sidebar	155
Troubleshooting	157
The stats helper	158
The popular images helper	161
Iterating by adding an image removal capability	162
Adding a route	162
Adding a controller handler	162
Updating the Handlebars image page template	163
Updating the jQuery	164
Refactoring and improvements	165
Summary	166
Chapter 8: Creating a RESTful API	167
What is an API?	168
What is a RESTful API?	168
Introducing Postman REST Client	169
Installation instructions	169
A quick tour of Postman REST Client	170
Using the JSONView Chrome extension	173
Creating a Basic API server	174
Creating sample JSON data	175
Responding to GET requests	176
Receiving data – POST and PUT requests	178

Removing data – DELETE	183
Consuming external APIs from Node.js	185
Consuming an API endpoint using Request	185
Summary	188
Chapter 9: Testing Your Code	189
Tools of the trade	189
Running tests with the Mocha framework	190
Asserting tests with Chai.js	192
Installing Chai.js as a devDependency	194
Spies and stubs with Sinon.js	194
Stubbing node modules with Proxyquire	197
Writing and running your first test	199
Writing a test helper	199
Testing the application	201
Testing the routes	202
Testing the server	204
Testing a model	207
Testing a controller	210
Spy and stub everything!	214
Summary	215
Chapter 10: Deploying with Cloud-based Services	217
Cloud versus traditional hosting	217
Infrastructure as a Service (IaaS) versus Platform as a Service (PaaS)	218
Introduction to Git	219
Deploying your application	220
Nodejitsu	220
Heroku	226
Amazon Web Services (AWS)	231
Create a MongoLab account and database	231
Create and configure the AWS environment	233
Microsoft Azure	236
Digital Ocean	242
Summary	244
Chapter 11: Single Page Applications with Popular Frontend Frameworks	245
What is a Single Page Application?	245
Why use a frontend framework?	246
The TodoMVC project	247
Backbone.js	248

Table of Contents

Ember.js	250
AngularJS	251
Frontend development tools	252
Automated build task managers	252
Dependency management	254
Modularity	255
HTML template-rendering engines	256
CSS transpiling	256
Testing and test-driven development	258
PhantomJS headless browser	258
Summary	259
Chapter 12: Popular Node.js Web Frameworks	261
Meteor	262
Sails	263
hapi	264
Koa	265
Flatiron	266
Summary	267
Index	269

Preface

My goal while writing *Web Development with MongoDB and Node.js* was simple: to empower you, the reader, with the tools and knowledge to be able to create web applications from scratch using Node.js and MongoDB.

In this book, we take a hands-on approach to building a complete, real-world, interactive web application. Each chapter will build upon the previous one, exposing new concepts, technologies, and best practices until finally ending with a completed application deployed to the cloud. Every line of code will be covered, and you are expected to code along with each chapter. Doing so will give you valuable insight into the world of web development using Node.js.

By the end of this book, I hope you have the expertise to tackle any project using Node.js and MongoDB and are limited only by your imagination!

What this book covers

Chapter 1, Welcome to JavaScript in the Full Stack, introduces you to the world of full stack JavaScript development and reviews what to expect in the remainder of the book.

Chapter 2, Getting Up and Running, walks you through the necessary steps to download, install, and configure your development environment.

Chapter 3, Node and MongoDB Basics, is a brief introduction to the basics of JavaScript, Node.js, and MongoDB.

Chapter 4, Writing an Express.js Server, introduces you to the Express.js Node.js Web Framework and is a walkthrough of the code necessary to write the main application server.

Chapter 5, Dynamic HTML with Handlebars, teaches you how to create dynamic HTML pages using Handlebars, the popular template-rendering engine.

Chapter 6, Controllers and View Models, walks you through writing the Controllers and View Models for the main application, the core of the application's functionalities.

Chapter 7, Persisting Data with MongoDB, continues with our Controllers and View Models, where we wrap all of the logic using Mongoose with MongoDB as the main data layer for the application.

Chapter 8, Creating a RESTful API, reviews the concepts behind REST APIs and introduces the Postman REST Client tool to test and interact with our own custom Node.js API.

Chapter 9, Testing Your Code, introduces the tools and techniques to write automated tests for our Node.js code.

Chapter 10, Deploying with Cloud-based Services, is a step-by-step walkthrough of deploying your application to a number of popular cloud-based hosting services such as Heroku, Microsoft Azure, and Amazon's AWS.

Chapter 11, Single Page Applications with Popular Frontend Frameworks, takes a look at the current trend in thick client applications by learning more about popular frontend single application frameworks such as Ember.js, AngularJS, and Backbone.js. Additionally, you will learn about the popular build tools frontend developers use to make their lives easier.

Chapter 12, Popular Node.js Web Frameworks, takes a look at some very popular and robust alternatives such as Meteor and Sails, even though Express.js is one of the most popular web frameworks for Node.

What you need for this book

In this book, the following software will be required:

- Operating systems:
 - Windows XP or superior
 - Mac OS X or superior
 - Linux

- Miscellaneous:
 - A standard text editor of choice
 - A web browser, preferably Google Chrome
- A command-line terminal of choice

Who this book is for

This book is designed for developers of any skill level that want to get up and running using Node.js and MongoDB to build full-featured web applications. A basic understanding of JavaScript and HTML is the only requirement for this book.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "Make sure you've npm installed all of the required modules for this chapter and that they are saved to your `package.json` file."

A block of code is set as follows:

```
models.Image.aggregate({ $group : {  
  _id : '1',  
  viewsTotal : { $sum : '$views' }  
}}, function(err, result) {  
  var viewsTotal = 0;  
  if (result.length > 0) {  
    viewsTotal += result[0].viewsTotal;  
  }  
  next(null, viewsTotal);  
});
```


When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:


```
.upload-button {  
  border-bottom: solid 2px #005A8B;  
  background: transparent $sprite-bg no-repeat;  
  @include radius(4px);  
  cursor: pointer;
```

Any command-line input or output is written as follows:

```
$ node server.js
Server up: http://localhost:3300
Mongoose connected.
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "Users demand more from their apps these days, and if you think about the application we've written, the **Like** button is a perfect example."

[ Warnings or important notes appear in a box like this.]

[ Tips and tricks appear like this.]

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to feedback@packtpub.com, and mention the book title via the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on www.packtpub.com/authors.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at <http://www.packtpub.com>. If you purchased this book elsewhere, you can visit <http://www.packtpub.com/support> and register to have the files e-mailed directly to you.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books – maybe a mistake in the text or the code – we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting <http://www.packtpub.com/submit-errata>, selecting your book, clicking on the **errata submission form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from <http://www.packtpub.com/support>.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at copyright@packtpub.com with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at questions@packtpub.com if you are having a problem with any aspect of the book, and we will do our best to address it.

1

Welcome to JavaScript in the Full Stack

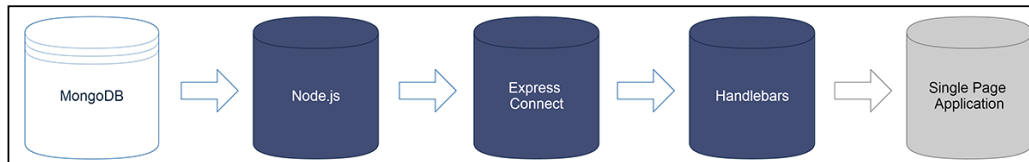
What an exciting time to be a JavaScript developer! What was once only considered a language to add enhancements and widgets to a webpage has since evolved into its own full-fledged ecosystem. I believe Atwood's law says it best— *any application that can be written in JavaScript, will eventually be written in JavaScript*. While this quote dates back to 2007, it's never been more true than today. Not only can you use JavaScript to develop a complete single-page web application such as Gmail, but you will also see how we can achieve the following projects with JavaScript throughout the remaining part of the book:

- How to completely power the backend using Node.js and Express.js
- How to persist data with a powerful database like MongoDB
- How to write dynamic HTML pages using Handlebars.js
- How to deploy your entire project to the cloud using services like Heroku and AWS

With the introduction of Node.js, JavaScript has officially gone in a direction that was never even possible before. Now, you can use JavaScript on the server, and you can also use it to develop full-scale enterprise-level applications. When you combine this with the power of MongoDB and its JSON-powered data, you can work with JavaScript in every layer of your application.

One of the great advantages of developing with JavaScript in the "full stack" of a web application is that you are using a consistent language and syntax. Frameworks and libraries are no longer exclusive only to the frontend or backend but can be integrated into other layers of the application as well.

Underscore.js is an extremely popular JavaScript library to work with collections that is used equally on the backend with Node.js as much as on the frontend directly within the browser.



JavaScript in the full stack of a web application

Node.js changed JavaScript forever

Back in 2009, Ryan Dahl gave a presentation at JSConf that changed JavaScript forever. During his presentation, he introduced Node.js to the JavaScript community, and after a roughly 45-minute talk, he concluded it, receiving a standing ovation from the audience in the process. He was inspired to write Node.js after he saw a simple file upload progress bar on Flickr, the image-sharing site. Realizing that the site was going about the whole process the wrong way, he decided that there had to be a better solution.

As stated on the Node.js homepage, the goal of Node is *to provide an easy way to build scalable network programs*. It achieves this by providing an event-driven, nonblocking IO model that is extremely lightweight. Compared to traditional web-serving technologies that require a new CPU thread for every connection to the server that would eventually max out the systems resources, Node instead uses a single thread but doesn't block the I/O of the CPU. Thus, this allows Node to support tens of thousands of concurrent connections. It's for this very reason that Node is so popular with high-traffic web applications.

To see an example of just how lightweight Node can be, let's take a look at some sample code that starts up an HTTP server and sends **Hello World** to a browser:

```
var http = require('http');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/plain'});
  res.end('Hello World\n');
}).listen(8080, 'localhost');
console.log('Server running at http://localhost:8080');
```

A few basic lines of code are all it takes to write a complete Node application. Running it with a simple `node app.js` command will launch an HTTP server that is listening on port 8080. Point any browser to `http://localhost:8080`, and you will see the simple output **Hello World** on your screen! While this sample app doesn't actually do anything useful, it should give you a glimpse of the kind of power you will have while writing web applications using Node.js.

At its core, Node is very low-level. It consists of a small set of modules that do very specific things and do them very well. These modules include tools to work with the file system, networking with TCP and HTTP, security, and streams.

Asynchronous callbacks

One of the most powerful features of Node is that it is event-driven and asynchronous. Code gets executed via callback functions whenever an event is broadcast. Simply put, you assign a callback function to an event, and when Node determines that the event has been fired, it will execute your callback function at that moment. No other code will get blocked waiting for an event to occur. Consider the following example to see asynchronous callbacks in action:

```
console.log('One');
console.log('Two');
setTimeout(function() {
  console.log('Three');
}, 2000);
console.log('Four');
console.log('Five');
```



Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at <http://www.packtpub.com>. If you purchased this book elsewhere, you can visit <http://www.packtpub.com/support> and register to have the files e-mailed directly to you.

In a typical synchronous programming language, executing the preceding code will yield the following output:

```
One
Two
... (2 second delay) ...
Three
Four
Five
```


However, in JavaScript and Node, the following output is seen:

```
One
Two
Four
Five
... (approx. 2 second delay) ...
Three
```

The function that actually logs Three is known as a callback to the `setTimeout` function.

Node Package Manager

Writing applications with Node is really enjoyable when you realize the sheer wealth of information and tools at your disposal! Using Node's built-in package manager *npm*, you can find literally tens of thousands of modules that can be installed and used within your application with just a few keystrokes! You can view the library of available modules by visiting <http://npmjs.org>. Downloading and installing any module within your application is as simple as executing the `npm install package` command. Have you written a module that you want to share with the world? Package it up using *npm*, and upload it to the public `npmjs.org` registry just as easily! Not sure how a module works that you downloaded and installed? The source code is right there in your projects' `node_modules/` folder waiting to be explored!

Networking and file IO

In addition to the powerful nonblocking asynchronous nature of Node, it also has very robust networking and filesystem tools available via its core modules. With Node's networking modules, you can create server and client applications that accept network connections and communicate via streams and pipes.

Not just on the web

Node isn't just for web development! It can be a powerful solution to create command-line tools as well as full-featured locally run applications that have nothing to do with the Web or a browser. Grunt.js is a great example of a Node-powered command-line tool that many web developers use daily to automate everyday tasks such as build processes, compiling CoffeeScript, launching Node servers, running tests, and more.

In addition to command-line tools, Node has recently become increasingly popular among the hardware crowd with the Nodebots movement. Johnny-Five and Cylon.js are two popular Node libraries that exist to provide a framework to work with robotics.

Real-time web with Socket.io

Node achieves real-time communication with Socket.io. Using Socket.io, you can create features such as instant collaboration, which is similar to multiuser editing in Google Docs. What was once achieved using cumbersome (and not real-time) long polling can now be achieved using WebSockets. While WebSockets is a feature that is only supported in modern browsers, Socket.io also features seamless fallback implementations for legacy browsers.

Using this lightweight core, everything else is left to the developer—but don't let that scare you. The beauty of working with Node is that there is a thriving community developing and releasing modules every day via npm. As of this writing, npm has over 61,000 packages available! Throughout this book, we will use some of the most popular packages that help make writing web applications fun and easy!

The NoSQL movement

The term *NoSQL* has come to mean any kind of database that doesn't adhere to the strict structures of a typical relational database such as Microsoft SQL, MySQL, PostgreSQL, and so on. With a relational database, you are required to define ahead of time the exact structure of your schema. This means that you must have defined the exact number of columns, length, and datatype for every field in a table, and that each field must always match that exact set of criteria.

With a NoSQL database server such as MongoDB, records are stored as JSON-like documents. A typical document (record) in a MongoDB collection (table) might look like the following code:

```
$ mongo
> db.contacts.find({email: 'jason@kroltech.com'}).pretty()

{
  "email" : "jason@kroltech.com",
  "phone" : "123-456-7890",
  "gravatar" : "751e957d48e31841ff15d8fa0f1b0acf",
  "_id" : ObjectId("52fad824392f58ac2452c992"),
  "name" : {
```

```
      "first" : "Jason",
      "last"  : "Krol"
    },
    "_v" : 0
  }
```

One of the biggest advantages of using a NoSQL database server such as MongoDB is that it has a dynamic schema system, allowing records in a collection to be completely different from one another.

Some advantages of working with MongoDB are:

- Dynamic schema design
- Fast querying and indexing
- Aggregate framework
- Sharding and replication

In addition, as MongoDB was written using a JSON-like document structure, JavaScript becomes a powerful tool when working with queries and the interactive shell *mongo*. Like Node, MongoDB is also built for high performance, making it a great counterpart for building ever demanding, high traffic web and mobile applications. Depending on your exact needs, MongoDB may or may not be the right solution for your application. You should truly weigh the pros and cons of each technology before making a decision to determine which technology is right for you.

Node and MongoDB in the wild

Both Node and MongoDB are extremely popular and active in the development community. This is true for enterprises as well. Some of the biggest names in the Fortune 500 space have fully embraced Node to power their web applications. This is due in large part to the asynchronous nature of Node, which makes it a great alternative for high traffic, high IO applications such as e-commerce websites and mobile applications.

The following is just a small list of some big companies that are working with Node:

- PayPal
- LinkedIn
- eBay
- Walmart
- Yahoo!

- Microsoft
- Dow Jones
- Uber
- New York Times

MongoDB's use in the enterprise sector is equally as impressive and wide reaching with an increasing number of companies adopting the leading NoSQL database server, such as:

- Cisco
- Craigslist Inc.
- Forbes
- FourSquare
- Intuit
- McAfee
- MTV
- MetLife
- Shutterfly
- Under Armour

What to expect from this book

The remainder of this book is going to be a guided tour that walks you through creating a complete data-driven website. The website we create will feature almost every aspect of a typical large-scale web development project. At its core, it will be powered by Node.js using a popular third-party framework called Express, and it will persist data using MongoDB.

In the first few chapters, we will cover the groundwork involved in getting the core of the server up and serving content. This includes configuring your environment so you are up and running with Node and MongoDB, and a basic introduction to the core concepts of both technologies. Then, we will write a web server from scratch powered by ExpressJS that will handle serving all of the necessary files for the website. From there, we will work with the Handlebars template engine to serve both static and dynamic HTML webpages. Diving deeper, we will make the application persistent by adding a data layer where the records for the website will be saved and retrieved via a MongoDB server. We will cover writing a RESTful API so that third parties can interact with your application. Finally, we will go into detail examining how to write and execute tests for all of your code.

Wrapping up, we will take a brief detour as we examine some popular, emerging frontend technologies that are becoming increasingly popular while writing single-page applications. These technologies include Backbone.js, Angular, and Ember.js.

Last but not least, we will go into details of how to deploy your new website to the Internet using popular cloud-based hosting services such as Heroku and Amazon Web Services.

Summary

In this chapter, we reviewed what is to be expected throughout the remainder of this book. We discussed the amazing current state of JavaScript and how it can be used to power the full stack of a web application. Not that you needed any convincing in the first place, but I hope you're excited and ready to get started writing web applications using Node.js and MongoDB!

Next up, we will set up your development environment and get you up and running with Node, MongoDB, and npm as well as write and launch a quick first Node app that uses MongoDB!

2

Getting Up and Running

The first thing you need to take care of is to make sure your development environment is equipped with the necessary requirements in order for you to use both Node and MongoDB while launching the apps you write.

In this chapter, we will cover the following topics:

- Installing and testing Node.js
- Installing, configuring, and testing MongoDB
- Writing and launching a simple app

Environment assumptions and requirements

For the remainder of this book, I will assume that you are using either a Mac with OS X, Linux, or Windows 7 or 8. You will also need superuser and/or administrator privileges on the computer, as you will be installing the Node and MongoDB server software. The code and examples after this chapter will all be OS agnostic and should work in any environment, assuming you have taken the steps I outline here so that you are prepared ahead of time.

You will need a good text editor to write and edit the code. Any editor of your liking will do. Personally, I am a huge fan of Sublime Text 3 (<http://sublimetext.com>). It is a simple, lightweight editor that has great color-coding syntax support. However, its true power comes from the unlimited plugins made available by other developers. There is literally a plugin for everything in Sublime! VI and Notepad are also good options if you want to stay super lightweight.