

# CONTENTS

*INTRODUCTION*

*xxvii*

## **PART I: INTRODUCTION AND SETUP**

### **CHAPTER 1: INSTALLING NODE 3**

<b>Installing Node on Windows</b>	<b>4</b>
<b>Installing on Mac OS X</b>	<b>5</b>
<b>Installing Node Using the Source Code</b>	<b>6</b>
Choosing the Node Version	6
Downloading the Node Source Code	6
Building Node	7
Installing Node	7
Running Node	8
<b>Setting Up and Using Node Package Manager</b>	<b>8</b>
Using NPM to Install, Update, and Uninstall Packages	9
<b>Summary</b>	<b>13</b>

### **CHAPTER 2: INTRODUCING NODE 15**

<b>Introducing the Event-Driven Programming Style</b>	<b>16</b>
<b>How Node and JavaScript Make Writing Asynchronous Applications Easier</b>	<b>17</b>
What Are Closures?	18
How Closures Help When Programming Asynchronously	19
<b>Summary</b>	<b>19</b>

## **PART II: NODE CORE API BASICS**

### **CHAPTER 3: LOADING MODULES 23**

<b>Understanding How Node Loads Modules</b>	<b>24</b>
<b>Exporting a Module</b>	<b>24</b>
Loading a Module	25
<b>Summary</b>	<b>28</b>

---

<b>CHAPTER 4: USING BUFFERS TO MANIPULATE, ENCODE, AND DECODE BINARY DATA</b>	<b>29</b>
Creating a Buffer	30
Getting and Setting Bytes in a Buffer	30
Slicing a Buffer	32
Copying a Buffer	32
Decoding a Buffer	32
Summary	33
<b>CHAPTER 5: USING THE EVENT EMITTER PATTERN TO SIMPLIFY EVENT BINDING</b>	<b>35</b>
Understanding the Standard Callback Pattern	36
Understanding the Event Emitter Pattern	36
Understanding Event Types	37
Using the Event Emitter API	38
Binding Callbacks Using <code>.addListener()</code> or <code>.on()</code>	38
Binding Multiple Event Listeners	39
Removing an Event Listener from an Event Emitter Using <code>.removeListener()</code>	40
Getting a Callback Executed at Most Once Using <code>.once()</code>	40
Removing All Event Listeners from an Event Emitter Using <code>.removeAllListeners()</code>	41
Creating an Event Emitter	41
Inheriting from Node Event Emitter	42
Emitting Events	42
Summary	43
<b>CHAPTER 6: SCHEDULING THE EXECUTION OF FUNCTIONS USING TIMERS</b>	<b>45</b>
Using <code>setTimeout</code> to Defer the Execution of a Function	46
Using <code>clearTimeout</code> to Cancel the Execution of a Function	46
Scheduling and Canceling the Repetitive Execution of a Function	47
Using <code>process.nextTick</code> to Defer the Execution of a Function Until the Next Event Loop Iteration	47
Blocking the Event Loop	48
Escaping the Event Loop	49
Using <code>setTimeout</code> Instead of <code>setInterval</code> to Force Serialization	49
Summary	50

---

**PART III: FILES, PROCESSES, STREAMS, AND NETWORKING**

<b>CHAPTER 7: QUERYING, READING FROM, AND WRITING TO FILES</b>	<b>53</b>
<b>Manipulating File Paths</b>	<b>54</b>
Normalizing Paths	54
Joining Paths	54
Resolving Paths	55
Finding the Relative Path Between Two Absolute Paths	55
Extracting Components of a Path	55
Determining the Existence of a Path	56
<b>Introducing the fs Module</b>	<b>57</b>
Querying File Statistics	57
<b>Opening a File</b>	<b>58</b>
<b>Reading from a File</b>	<b>59</b>
Writing to a File	60
Closing a File	60
<b>Summary</b>	<b>62</b>
<b>CHAPTER 8: CREATING AND CONTROLLING EXTERNAL PROCESSES</b>	<b>63</b>
<b>Executing External Commands</b>	<b>64</b>
<b>Spawning Child Processes</b>	<b>69</b>
Creating the Child Process	69
Listening for Data from the Child Process	69
Sending Data to the Child Process	70
Receiving Notification When the Child Process Exits	72
<b>Signaling and Killing Processes</b>	<b>73</b>
<b>Summary</b>	<b>74</b>
<b>CHAPTER 9: READING AND WRITING STREAMS OF DATA</b>	<b>75</b>
<b>Using a Readable Stream</b>	<b>76</b>
Waiting for Data	76
Pausing and Resuming a Stream	77
Knowing When the Stream Ends	77
<b>Using Writable Streams</b>	<b>77</b>
Writing Data into a Stream	78
Waiting for a Stream to Drain	78
<b>Considering Some Stream Examples</b>	<b>78</b>
Creating File-System Streams	79
Understanding Networking Streams	80

---

<b>Avoiding the Slow Client Problem and Saving Your Server</b>	<b>80</b>
Understanding the Slow Client Problem	80
Avoiding the Slow Client Problem	81
Using stream.pipe() to Prevent the Slow Client Problem and Assembling Readable and Writable Streams Using pipe()	82
<b>Summary</b>	<b>82</b>
<b>CHAPTER 10: BUILDING TCP SERVERS</b>	<b>83</b>
<hr/>	
<b>Creating a TCP Server</b>	<b>83</b>
Using the Socket Object	85
Understanding Idle Sockets	86
Setting Up Keep-Alive	87
Using Delay or No Delay	87
Listening for Client Connections	88
Closing the Server	88
Handling Errors	88
<b>Building a Simple TCP Chat Server</b>	<b>89</b>
Accepting Connections	89
Reading Data from a Connection	90
Collecting All the Clients	90
Broadcasting Data	91
Removing Closed Connections	92
Using Your TCP Chat Server	93
<b>Summary</b>	<b>93</b>
<b>CHAPTER 11: BUILDING HTTP SERVERS</b>	<b>95</b>
<hr/>	
<b>Understanding the http.ServerRequest Object</b>	<b>97</b>
<b>Understanding the http.ServerResponse Object</b>	<b>98</b>
Writing a Header	98
Changing or Setting a Header	99
Removing a Header	99
Writing a Piece of the Response Body	99
<b>Streaming HTTP Chunked Responses</b>	<b>99</b>
Piping a File	100
Piping the Output of Another Process	100
<b>Shutting Down the Server</b>	<b>101</b>
<b>Example 1: Building a Server that Serves Static Files</b>	<b>101</b>
<b>Example 2: Making Use of HTTP Chunked Responses and Timers</b>	<b>102</b>
<b>Summary</b>	<b>102</b>

---

<b>CHAPTER 12: BUILDING A TCP CLIENT</b>	<b>103</b>
<b>Connecting to a Server</b>	<b>104</b>
<b>Sending and Receiving Data</b>	<b>105</b>
<b>Ending the Connection</b>	<b>105</b>
<b>Handling Errors</b>	<b>106</b>
<b>Building an Example Command-Line TCP Client</b>	<b>106</b>
Connecting to the Server	107
Sending the Command Line to the Server	107
Printing Server Messages	107
Reconnecting if the Connection Dies	108
Closing the Connection	109
Putting the Client Together	111
<b>Summary</b>	<b>112</b>
<b>CHAPTER 13: MAKING HTTP REQUESTS</b>	<b>113</b>
<b>Making GET Requests</b>	<b>113</b>
<b>Using Other HTTP Verbs</b>	<b>114</b>
Inspecting the Response Object	115
Obtaining the Response Body	116
Streaming the Response Body	116
<b>Pooling Sockets Using http.Agent</b>	<b>116</b>
<b>Using a Third-Party Request Module to Simplify HTTP Requests</b>	<b>118</b>
Installing and Using Request	118
Creating a Testing Server	120
Following Redirects	121
Setting Some Request Options	123
Encoding the Request Body	125
Streaming	127
Using a Cookie Jar	127
<b>Summary</b>	<b>128</b>
<b>CHAPTER 14: USING DATAGRAMS (UDP)</b>	<b>129</b>
<b>Understanding UDP</b>	<b>129</b>
<b>Understanding the Uses of UDP</b>	<b>130</b>
<b>Building a Datagram Server</b>	<b>130</b>
Listening for Messages	130
Testing the Server	131
Inspecting Additional Message Information	132
<b>Creating a Simple Datagram Echo Server</b>	<b>132</b>

---

Waiting for Messages	132
Sending Messages Back to Senders	132
Putting the Echo Server Together	133
<b>Building a Datagram Client</b>	<b>134</b>
Creating the Client	134
Sending Messages	134
Closing the Socket	134
<b>Creating a Simple Datagram Command-Line Client</b>	<b>135</b>
Reading from the Command Line	135
Sending Data to the Server	135
Receiving Data from the Server	136
Putting the Command-Line UDP Client Together	136
<b>Understanding and Using Datagram Multicast</b>	<b>136</b>
Receiving Multicast Messages	137
Sending Multicast Messages	138
Understanding Maximum Datagram Size	138
<b>Summary</b>	<b>138</b>
<b>CHAPTER 15: SECURING YOUR TCP SERVER WITH TLS/SSL</b>	<b>139</b>
<hr/>	
<b>Understanding Private and Public Keys</b>	<b>139</b>
Generating a Private Key	140
Generating a Public Key	140
<b>Building a TLS Server</b>	<b>141</b>
Initializing the Server	141
Listening for Connections	141
Reading Data from the Client	142
Sending Data to the Client	142
Ending the Connection	142
<b>Building a TLS Client</b>	<b>143</b>
Initializing the Client	143
Connecting to the Server	143
Verifying the Server Certificate	143
Sending Data to the Server	144
Reading Data from the Server	144
Ending the Connection	144
<b>Building Some Examples</b>	<b>145</b>
Creating a TLS Chat Server	145
Creating a TLS Command-Line Chat Client	146
Verifying the Client Certificate	147
<b>Summary</b>	<b>148</b>

---

<b>CHAPTER 16: SECURING YOUR HTTP SERVER WITH HTTPS</b>	<b>149</b>
<b>Building a Secure HTTP Server</b>	<b>149</b>
Setting Up the Server Options	150
Listening for Connections	150
Validating the HTTPS Client Certificate	151
<b>Creating an HTTPS Client</b>	<b>152</b>
Initializing the Client	152
Making the Request	152
Validating the HTTPS Server Certificate	153
<b>Summary</b>	<b>154</b>
<b>PART IV: BUILDING AND DEBUGGING MODULES AND APPLICATIONS</b>	
<b>CHAPTER 17: TESTING MODULES AND APPLICATIONS</b>	<b>157</b>
<b>Using a Test Runner</b>	<b>157</b>
Writing Tests	158
Running Tests	159
<b>Using an Assertion Testing Module</b>	<b>159</b>
Using the assert Module	159
Using the Built-in Assertion Functions in Node-Tap	161
<b>Testing Your Asynchronous Module</b>	<b>163</b>
<b>Summary</b>	<b>166</b>
<b>CHAPTER 18: DEBUGGING MODULES AND APPLICATIONS</b>	<b>167</b>
<b>Using console.log</b>	<b>167</b>
<b>Using Node's Built-in Debugger</b>	<b>169</b>
<b>Using Node Inspector</b>	<b>173</b>
<b>Summary</b>	<b>175</b>
<b>CHAPTER 19: CONTROLLING THE CALLBACK FLOW</b>	<b>177</b>
<b>Understanding the Boomerang Effect</b>	<b>177</b>
<b>Avoiding the Boomerang Effect by Declaring Functions</b>	<b>179</b>
<b>Using the async Flow Control Library</b>	<b>183</b>
Executing in Series	184
Executing in Parallel	185
Cascading	186
Queuing	187
Iterating	189
Mapping	190

Reducing	191
Filtering	192
Detecting	193
<b>Summary</b>	<b>194</b>

## **PART V: BUILDING WEB APPLICATIONS**

### **CHAPTER 20: BUILDING AND USING HTTP MIDDLEWARE** 197

<b>Understanding the Connect HTTP Middleware Framework</b>	<b>198</b>
<b>Building Your Own HTTP Middleware</b>	<b>198</b>
Creating Asynchronous Middleware	200
Registering Callbacks Inside Middleware	201
Handling Errors Inside Middleware	203
<b>Using the HTTP Middleware Bundled in Connect</b>	<b>206</b>
Logging Requests	206
Handling Errors	208
Serving Static Files	209
Parsing the Query String	210
Parsing the Request Body	211
Parsing Cookies	212
Using a Session	213
Other Available Middleware	216
<b>Summary</b>	<b>216</b>

### **CHAPTER 21: MAKING A WEB APPLICATION USING EXPRESS.JS** 217

<b>Initializing Your Express.js Application</b>	<b>218</b>
<b>Setting Up Middleware in Your Application</b>	<b>220</b>
<b>Routing Requests</b>	<b>222</b>
Handling Routes	222
Using Sessions	229
Using Route Middleware	234
<b>Summary</b>	<b>238</b>

### **CHAPTER 22: MAKING UNIVERSAL REAL-TIME WEB APPLICATIONS USING SOCKET.IO** 241

<b>Understanding How WebSockets Work</b>	<b>242</b>
<b>Using Socket.IO to Build WebSocket Applications</b>	<b>243</b>
Installing and Running Socket.IO on the Server	243
Building a Real-Time Web Chat with Socket.IO	245
Extending the Chat Application	250
Detecting Disconnections	254



Separating Users into Rooms	255
Using Namespaces	259
Distributing the Server-Side Application Using Redis	259
<b>Summary</b>	<b>263</b>

## **PART VI: CONNECTING TO DATABASES**

### **CHAPTER 23: CONNECTING TO MYSQL USING NODE-MYSQL**      **267**

Using a Library to Connect to and Communicate with a MySQL Database	268
Adding Data to the Database with Security Concerns in Mind	270
Reading Data Efficiently	272
<b>Summary</b>	<b>276</b>

### **CHAPTER 24: CONNECTING TO COUCHDB USING NANO**      **277**

Installing Nano	278
Connecting and Creating a Database	281
Storing Documents	285
Creating and Using CouchDB Views	286
Attaching Files to a CouchDB Document	298
<b>Summary</b>	<b>310</b>

### **CHAPTER 25: CONNECTING TO MONGODB USING MONGOOSE**      **311**

Installing Mongoose	313
Understanding How Mongoose Uses Models to Encapsulate Database Access	313
Connecting to MongoDB	314
Defining a Schema	314
Defining a Model	315
Using Validators	324
Using Modifiers	330
Using Getters	331
Using Virtual Attributes	332
Using Default Values	338
Defining Indexes	340
Referencing Other Documents Using DB Refs	341
Defining Instance Methods	347
Defining Static Methods	348
<b>Summary</b>	<b>349</b>

### **INDEX**      **351**