

contents

<i>preface</i>	xiii
<i>acknowledgments</i>	xv
<i>about this book</i>	xviii

PART 1 BASICS OF THE IOT AND THE WOT 1

1 *From the Internet of Things to the Web of Things* 3

1.1 Defining the Internet of Things 4

1.2 Enter the Web of Things 6

Web of Things scenario: connected hotel 6

Comparing IoT and WoT 8 ▪ *The Internet of Things—
a brief history* 12

1.3 Use cases—why connected objects? 14

Wireless sensor networks and distributed

sensing 14 ▪ *Wearables and quantified self* 16 ▪ *Smart
homes and buildings* 17 ▪ *Smart cities and energy grids* 18
Smart manufacturing and Industry 4.0 19 ▪ *Smart logistics
and supply chains* 19 ▪ *Marketing 2.0* 21

- 1.4 The Web of Things—a supercharged Internet of Things 22
 - Easier to program* 23 ▪ *Open and extensible standards* 24
 - Fast and easy to deploy, maintain, and integrate* 24
 - Loose coupling between elements* 25 ▪ *Widely used security and privacy mechanisms* 25 ▪ *WoT—the shortcomings* 26
- 1.5 Summary 27

2 Hello, World Wide Web of Things 29

- 2.1 Meet a Web of Things device 30
 - The suspect: Raspberry Pi* 31
- 2.2 Exercise 1—Browse a device on the Web of Things 32
 - Part 1—The web as user interface* 32 ▪ *Part 2—The web as an API* 36 ▪ *So what?* 41
- 2.3 Exercise 2—Polling data from a WoT sensor 42
 - Part 1—Polling the current sensor value* 42 ▪ *Part 2—Polling and graphing sensor values* 43 ▪ *Part 3—Real-time data updates* 44 ▪ *So what?* 45
- 2.4 Exercise 3—Act on the real world 46
 - Part 1—Use a form to update text to display* 46 ▪ *Part 2—Create your own form to control devices* 48 ▪ *So what?* 50
- 2.5 Exercise 4—Tell the world about your device 50
 - So what?* 54
- 2.6 Exercise 5—Create your first physical mashup 54
 - So what?* 57
- 2.7 Summary 57

3 Node.js for the Web of Things 59

- 3.1 The rise of JavaScript: from clients to servers to things! 60
 - Pushing JavaScript to things* 61
- 3.2 Introduction to Node.js 62
 - Installing Node.js on your machine* 63 ▪ *Your first web server in Node.js* 63 ▪ *Returning sensor data as JSON* 65
- 3.3 Modularity in Node.js 66
 - npm—the Node package manager* 67 ▪ *Clean dependencies with package.json and npm* 68 ▪ *Your first Node module* 69

- 3.4 Understanding the Node.js event loop 71
 - Multithreaded web servers* 71 ▪ *Single-threaded, non-blocking web servers* 71
- 3.5 Getting started with asynchronous programming 73
 - Anonymous callbacks* 74 ▪ *Named callbacks* 77 ▪ *Control flow libraries* 78
- 3.6 Summary and beyond the book 81

4 **Getting started with embedded systems** 83

- 4.1 The world of embedded devices 84
 - Devices for hobbyists vs. industrial devices* 84 ▪ *Real-time operating systems vs. Linux* 85 ▪ *Summary and beyond the Pi* 87
- 4.2 Set up your first WoT device—Raspberry Pi 88
 - Meet the Raspberry Pi* 88 ▪ *Choosing your Pi* 89
 - Shopping list* 90 ▪ *Setting up your Raspberry Pi* 91
 - Connecting to your device* 96
- 4.3 Installing Node.js on the Raspberry Pi 97
 - Using Git and GitHub on the Pi* 99 ▪ *So what?* 99
- 4.4 Connecting sensors and actuators to your Pi 99
 - Understanding GPIO ports* 100 ▪ *Working with breadboards and electronic components* 100 ▪ *Accessing GPIOs from Node.js* 102 ▪ *Beyond the book* 107
- 4.5 Summary 108

5 **Building networks of Things** 109

- 5.1 Connecting Things 111
 - Network topologies* 111 ▪ *Network classification models* 113
- 5.2 Networking protocols for Things 115
 - Spatial considerations* 115 ▪ *Internet protocols and the IoT* 116 ▪ *IoT personal area networks* 120 ▪ *IoT wide area networks* 124 ▪ *So, which one should I choose?* 127
- 5.3 Application protocols for Things 130
 - ZigBee and Bluetooth application stacks* 130 ▪ *Apple HomeKit and Google Weave* 132 ▪ *Message Queuing Telemetry Transport* 132 ▪ *Constrained Application Protocol* 135
 - So, which one should I use?* 135

5.4 The Web of Things architecture 136

Layer 1: Access 136 ▪ Layer 2: Find 137 ▪ Layer 3: Share 138 ▪ Layer 4: Compose 138 ▪ Why does the WoT matter? 138 ▪ Beyond the book 139

5.5 Summary 140

PART 2 BUILDING THE WOT..... 141

6 Access: Web APIs for Things 143

6.1 Devices, resources, and web Things 144

Representational State Transfer 144 ▪ Why do we need a uniform interface? 146 ▪ Principle 1: addressable resources 147 ▪ Principle 2: manipulation of resources through representations 151 ▪ Principle 3: self-descriptive messages 154 ▪ Principle 4: Hypermedia as the Engine of Application State 160 ▪ Summary—web Things design process 163

6.2 Beyond REST: the real-time Web of Things 163

The WoT needs events! 164 ▪ Publish/subscribe 165 Webhooks—HTTP callbacks 166 ▪ Comet—hacking HTTP for a real-time web 167 ▪ WebSockets 168 ▪ The future: from HTTP/1.1 to HTTP/2 172

6.3 Summary 173

7 Implementing web Things 175

7.1 Connecting devices to the web 176

7.2 Direct integration pattern—REST on devices 177

Creating a WoT server 178 ▪ Resource design 180 Representation design 186 ▪ Interface design 189 Pub/sub interface via WebSockets 191 ▪ Summary—direct integration pattern 194

7.3 Gateway integration pattern—CoAP example 194

Running a CoAP server 195 ▪ Proxying CoAP via a gateway 196 ▪ Summary—gateway integration pattern 198

7.4 Cloud integration pattern—MQTT over EVERYTHING 199

Set up your EVERYTHING account 201 ▪ Create your MQTT client application 204 ▪ Use actions to control the power

	<i>plug</i>	206	▪	<i>Create a simple web control application</i>	208
	<i>Summary—cloud integration pattern</i>	212			
7.5	Summary	212			
8	Find: Describe and discover web Things	214			
8.1	The findability problem	215			
8.2	Discovering Things	217			
	<i>Network discovery</i>	217	▪	<i>Resource discovery on the web</i>	220
8.3	Describing web Things	223			
	<i>Introducing the Web Thing Model</i>	225	▪	<i>Metadata</i>	227
	<i>Properties</i>	227	▪	<i>Actions</i>	229
	<i>Things</i>	231			
	<i>Implementing the Web Thing Model on the Pi</i>	232			
	<i>Summary—the Web Thing Model</i>	238			
8.4	The Semantic Web of Things	239			
	<i>Linked data and RDFa</i>	239	▪	<i>Agreed-upon semantics:</i>	
	<i>Schema.org</i>	243	▪	<i>JSON-LD</i>	244
	<i>Beyond the book</i>	246			
8.5	Summary	247			
9	Share: Securing and sharing web Things	248			
9.1	Securing Things	250			
	<i>Encryption</i>	101	252	▪	<i>Web security with TLS: the S of HTTPS!</i>
	<i>Enabling HTTPS and WSS with TLS on your Pi</i>	255			
9.2	Authentication and access control	260			
	<i>Access control with REST and API tokens</i>	260	▪	<i>OAuth: a web authorization framework</i>	263
9.3	The Social Web of Things	265			
	<i>A Social Web of Things authentication proxy</i>	266			
	<i>Implementing a Social WoT authentication proxy</i>	269			
9.4	Beyond the book	276			
9.5	Summary	278			
10	Compose: Physical mashups	279			
10.1	Building a simple app—automated UI generation	280			
	<i>A universal user interface for web Things</i>	281			
10.2	Physical mashups	288			
	<i>Boxes and wires mashups for the Physical Web: Node-RED</i>	289			

10.3	Using wizards for physical mashups: IFTTT	295
	<i>Pushing intruder alert tweets to a Google spreadsheet</i>	296
	<i>Sending requests to a Thing with the Maker Channel</i>	298
	<i>Pushing intruder alert tweets to a Google spreadsheet</i>	299
10.4	Beyond the book	300
	<i>From simple mashups to big data mashups</i>	300
	<i>A better user experience</i>	301
10.5	Summary	302
appendix	Arduino, BeagleBone, Intel Edison, and the WoT	303
	index	309