Community Experience Distilled

Learning Internet of Things

Explore and learn about Internet of Things with the help of engaging and enlightening tutorials designed for the Raspberry Pi

Peter Waher

PACKT PUBLISHING

Learning Internet of Things

Explore and learn about Internet of Things with the help of engaging and enlightening tutorials designed for Raspberry Pi

Peter Waher

[PACKT]
PUBLISHING

BIRMINGHAM - MUMBAI

Learning Internet of Things

First published: January 2015

Production reference: 1210115

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

ISBN 978-1-78355-353-2

www.packtpub.com

Credits

Author
Peter Waher

Reviewers
Fiore Basile
Dominique Guinard
Phodal Huang
Joachim Lindborg
Ilesh Patel

Commissioning Editor
Akram Hussain

Acquisition Editors
Richard Brookes-Bland
Richard Harvey

Content Development Editor
Anila Vincent

Technical Editor
Anushree Arun Tendulkar

Copy Editors
Gladson Monteiro
Jasmine Nadar

Project Coordinator
Neha Bhatnagar

Proofreaders
Ameesha Green
Amy Johnson

Indexer
Hemangini Bari

Graphics
Sheetal Aute
Valentina D'silva

Production Coordinator
Manu Joseph

Cover Work
Manu Joseph

About the Author

Peter Waher is the cofounder of Clayster, a company with its origin in Scandinavia but now operates in four continents. Clayster is dedicated to the development of Internet of Things applications and provides an IoT platform for rapid application development. Currently, Peter lives and works in Chile where he is the CEO of Clayster Laboratorios Chile S.A., a subsidiary of Clayster that provides development expertise to partner companies and promotes the Internet of Things technology to research institutions. Originally a mathematician, commercial pilot, and computer games developer, he has worked for 20 years with computers and device communication, including low-level development in assembler for resource-constrained devices to high-level system design and architecture. He's currently participating in various standardization efforts within IEEE, UPnP, and XSF, working on designing standards for Internet of Things. His work on smart applications for Internet of Things and the development of the IP-TV application "Energy Saving through Smart Applications" won the Urban Living Labs global showcase award in the Cultural and Societal Participation and Collaboration Tools category. Peter Waher can be found on LinkedIn at `http://linkedin.com/in/peterwaher/`.

I'd like to thank the founder of Clayster, Rikard Strid, and Packt Publishing for the opportunity to write this book; Joachim Lindborg for the many ideas and discussions related to Internet of Things; Fernando Cruz and Freddy Jimenez for their invaluable help with many practical details; my eldest daughter, Maria-Lorena, for accepting to stand model and offer to break into my office at night; and finally my wife and children for tolerating the many late hours it took to write this book.

About the Reviewers

Fiore Basile is a programmer, system administrator, creative, entrepreneur and maker. Since 1996, he has served as project manager, consultant, and technology officer in industrial and research projects of many sizes across Italy and Europe. He worked in the fields of cultural heritage, e-health, digital preservation, multimodal interfaces, web and mobile publishing. During his career, he also founded two IT start-ups, held workshops at international conferences and events, and has been interviewed by national and international press. His work experience allowed him to build a broad expertise in systems, web and mobile software development, open source and open hardware, embedded programming, and electronics. He's currently conducting research on wearable technologies, effective computing, and smart connected devices, and he is working as the coordinator of FabLab Cascina, a digital fabrication laboratory in the middle of Tuscany.

Dominique "Dom" Guinard is the CTO and cofounder of EVRYTHNG, a Web of Things and Internet of Things software company that makes products smart by connecting them to the Web. Dom got his PhD from ETH Zurich where he worked on defining the Web of Things architecture, a worldwide network of interconnected objects (sensor networks, appliances, machines, and tagged objects). He also cofounded `WebofThings.org` and the Web of Things conference series.

Before this, he worked on bringing industrial networks of RFID-tagged objects and smart things to the Web at the MIT Auto-ID Labs and was a visiting researcher at the MIT Mobile Experience Lab. He also worked for 4 years with SAP on designing scalable software architectures and infrastructures to integrate real-world objects with business systems. Dom was a researcher at the Auto-ID Labs, Zurich, where he worked on using mobile phones as gateways to Internet of Things (IoT) for Nokia. Before this, he worked on scalable IoT enterprise software architectures for RFID and embedded devices in collaboration with Sun Microsystems.

He holds an MSc degree in computer science and a BSc in computer science and management with a specialization in mobile and ubiquitous computing. In 2011, Dominique was listed fifth among the world's top 100 IoT thinkers. Early in 2012, his PhD research on the Web of Things was awarded the ETH Medal.

Phodal Huang has over 4 years of experience in hardware and web development. He graduated from Xi'an University of Arts and Science. He currently works at ThoughtWorks as a developer. He is the owner of the mini IoT project (`https://github.com/phodal/iot`) and the author of the eBook, *Design IoT* (`http://designiot.phodal.com`, in Chinese). He loves designing, painting, writing, traveling, and hacking; you can find out more about him on his personal website at `http://www.phodal.com`.

Joachim Lindborg is a dedicated systems engineer with a long experience of all the technologies that have been passed through the years, starting from Texas Instrument TI-16 to deploying Docker components on Core-Os on a distributed network of Intel NUC machines.

He is deeply into the exploding area of small devices. Electronics has always been fascinating and Joachim started soldering electronics in seventh grade. The Raspberry explosion with open hardware and software and MakerSpace enthusiasm is a revival and reclaim from the big producers.

Joachim's current focus is to combine these different forms of knowledge of large systems and hardware with meters and actuators to create smart energy services.

Starting in 1993, Joachim was part of the biggest telecom project in Ericsson. The project aimed at creating the next century telecom platforms, ATM. TCP/IP seems to be the winner. For his next big project, he was at the Swedish Utility for several years, building smart home services, which was a pre-millennium shift as they were using phone lines and modems. This is a dead technology now.

It was really in 2002, when Joachim was one of the founders of `homesolutions.se.loopiadns.com`, that his system architect skills were used to create a distributed Linux system. Today, this system has some 46,000 apartments that measure electricity, water, and so on, and create advanced building automation.

In his current assignment as the CTO for sustainable innovation, there is a constant need for IoT-distributed logic and advanced data analyses to gain energy efficiency and a sustainable society.

He has also contributed to a Swedish IT architect book, `http://www.thearchitectbook.com/`.

Ilesh Patel holds a bachelor's degree in electronics and communication and a master's degree in VLSI and Embedded System Design. He has more than 3 years of experience as an embedded engineer. He has good debugging skills and command over the high-level C/C++ language, the scripting language Python, and the hardware language VHDL. He has knowledge and hands-on experience on how to design and develop an automated test suite framework using Python, Microcontroller, and an FPGA-based system design development.

I'd like to thank my friend Uchit Vyas for encouraging me to review this book.

Table of Contents

Preface

Internet of Things is one of the current top tech buzzwords. Large corporations value its market in tens of trillions of dollars for the upcoming years, investing billions into research and development. On top of this, there is the plan for the release of tens of billions of connected devices during the same period. So you can see why it is only natural that it causes a lot of buzz.

Despite this, nobody seems to agree on what Internet of Things (IoT) actually is. The only thing people agree on is that whatever it is, it is worth a lot of money. And where there is a lot of money, there is a lot of competition, which in reality means a lot of confusion. To be able to stand out as a superior player, companies invent new buzz words in an attempt to highlight their superior knowledge. In this battle of gaining the reader's attention, the world is now seeing a plethora of new definitions, one better than the other, such as "Internet of Everything," "Web of Things," "Internet of People and Things," and so on. To pour gasoline on fire, there is a constant overlap and confusion of ideas from related terms, such as "Big Data," "Machine-to-Machine," and "Cyber-Physical Systems" to mention a few.

This lack of consensus on what IoT actually is and what it means makes it somewhat difficult to write a book on the subject. Not because the technical aspects are difficult—they are not—but because you need to define what it is you are going to talk about and also what you are not going to talk about. You need to define IoT in a way that is simple, valid, and constructive, while at the same time it should minimize controversy.

A definition for Internet of Things

To be able to define IoT, let's first look at how the term was coined. Kevin Ashton noted that most data on the Internet was at the time originally entered or captured into the system by human beings. From a system point of view, a human is nothing more than a slow, error-prone, and inefficient router of data that puts limits on quality and quantity of data available and sometimes even dares to interpret data or correct it. As an alternative, it would be more efficient if these systems could connect to sensors that measure these real-world events or properties directly. So, in this vision, systems bypass human intermediaries and connect directly to sensors connected to the Internet to capture real-world data.

The problem with this definition is that it is not a definition at all but a vision, albeit with an important point. If systems can access data captured by sensors directly, of course, the data will be both more abundant and more correct. This was known decades ago and is a field of study in its own right, labeled "sensor networks". What is the real difference between these two? What is the difference between IoT and Big Data, where the efficient storage of huge volumes of data is handled? How does IoT differ from machine-to-machine (M2M) or device-to-device (D2D) communication, where communication between Things is discussed? Or, how does it differ from cyber-physical systems (CPS) that concerns itself with systems that interact with the real world through sensors and actuators? What is the real difference between IoT and the just mentioned fields of study?

Let's, therefore, have a very simple definition and see where it leads us:

The IoT is what we get when we connect Things, which are not operated by humans, to the Internet.

Competing definitions

IoT is not the same as sensor networks since Things neither need to be sensors, nor do sensor networks need to be connected to the Internet. Also, IoT is not the same as big data since neither Things are required to capture or generate data, nor do applications need to store the data centrally (in the Cloud) in big data stores. IoT is not part M2M since being on the Internet implies humans can (and want to) access these Things directly too. Furthermore, the latter, as well as CPS, also concern themselves with non-Internet protocols, transport of messages between machines and/or devices in the network, as well as automation, often in closed and controlled environments.

Being connected to the Internet is much more than simple connectivity and message transport. The Internet is open, meaning anybody can add Things to it. It also means they will want Things to interoperate in a loosely coupled manner. The Internet is not only open, but it also is the largest network in the world. It is also the foulest cesspit in the world. Connect something to the Internet, and you can be rest assured that somebody will try to take advantage of it or destroy it if they can, just for the sheer fun of it. Comparing IoT to M2M communication is like assuming that an experiment in a controlled laboratory environment will continue to work even when you let a bunch of 3-year-old kids high on caffeinated beverages enter the laboratory, equipped with hammers and a playful attitude, and promised ice cream if they destroy everything they could see.

While some are concerned that IoT is too limited to include people in the equation, and it invents new terminologies such as Internet of People and Things, this is already included in the definition we just saw where we noted that people are already connected to the Internet via computers when we connect Things. Such a definition is therefore not necessary. Others discuss a Web of Things (WoT), which is a subset of IoT, where communication is limited to web technologies, such as HTTP, browsers, scripting, and so on. This view might stem from equaling the Internet with the World Wide Web (WWW), where access to the Internet is made through browsers and URLs. Even though we will discuss web technologies in this book, we consider web technologies alone too limiting.

There are also misleading definitions that act more like commercial buzz words rather than technological terminology, such as Internet of Everything, promoting the idea of being something more than IoT. But what is included in Internet of Everything that is not already included in IoT? All connectable Things are already included in IoT. Things that cannot be connected directly (air or water), or indirectly (vacuum or happiness) cannot be accessed in Internet of Everything either, just because the name says so. Everything needs a Thing or a Person to connect to the Internet. There are claims that the Internet of Everything includes processes, and such, and would differ in that sense. But, in the definition we just saw, such processes would be simple corollaries and require no new definition.

Direct consequences

Now that we have a clear definition of IoT, as something we get when we connect Things, not operated by humans, to the Internet, we are ready to begin our study of the subject. The definition includes four important components:

- Connection, which relates to the study of communication protocols

- Things, which relates to the study of sensors, actuators, and controllers, among other Things
- Non-operation by humans relates to provisioning
- The Internet relates to security, including identities, authentication, and authorization, but also to interoperability

This book will introduce all these concepts one at a time using simple and practical examples illustrating how these key concepts can be implemented using the Raspberry Pi platform.

What this book covers

Chapter 1, *Preparing our IoT Projects*, introduces the projects we will use throughout the book, their basic project structure, our development environment, how to prepare our Raspberry Pi, and how to perform basic input and output operations on it.

Chapter 2, *The HTTP Protocol*, presents the basics of the HTTP protocol and how it can be used in IoT applications. It also describes how it relates to the request/response and event subscription communication patterns.

Chapter 3, *The UPnP Protocol*, presents the basics of the UPnP protocol and how it can be used to discover devices in an ad hoc local area network. It also discusses how to call services on the devices and subscribe to events from them. Additionally, it describes how to build devices that publish such discoverable services and events.

Chapter 4, *The CoAP Protocol*, presents the basics of the CoAP protocol and how it can be used on devices that communicate over bandwidth-limited networks. It will show you how to publish content, how to subscribe to events, how to transport large content items using blocks, and how to discover existing resources on a device.

Chapter 5, *The MQTT Protocol*, introduces the MQTT protocol and shows how our IoT applications can bypass firewalls using the publish/subscribe communication pattern and message brokers.

Chapter 6, *The XMPP Protocol*, presents the XMPP protocol, and how it uses a federated set of message brokers to provide global identities and how it provides a richer set of communication patterns to the IoT developer. This chapter also introduces new communication patterns such as friendship authorization, presence subscription, asynchronous messaging, delegation of trust, and provisioning.

Chapter 7, *Using an IoT Service Platform*, presents reasons for using a service platform designed for IoT to facilitate rapid application development of IoT services and taking care of recurrent issues, such as security, interoperability, scalability, management, monitoring, and so on.

Chapter 8, *Creating Protocol Gateways*, shows you how to use good abstraction models to facilitate the creation of protocol bridges, allowing the interconnection of systems and services based on different technologies. This will enable you to design a secure and interoperable infrastructure for smart cities based on the IoT.

Chapter 9, *Security and Interoperability*, gives an overview of available threats and common modes of attack and how to build counter measures to protect your IoT solutions. It also shows the importance of interoperability in IoT and how to avoid limiting one to favor the other.

Appendix A, *Console Applications*, shows the basic structure of console applications, as used throughout the examples in this book.

Appendix B, *Sampling and History*, shows how sampling and historical record keeping of sensor values is done in the Sensor project published in this book.

Appendix C, *Object Database*, shows how to persist data in an object database, simply by using class definitions.

Appendix D, *Control*, shows how control operations are implemented in the Actuator project published in this book.

Appendix E, *Fundamentals of HTTP*, provides an overview of the fundamentals of the HTTP protocol.

Appendix F, *Sensor Data Query Parameters*, provides a set of query parameters we can use to limit readout requests from devices to data we are interested in.

Appendix G, *Security in HTTP*, discusses different ways to implement security into applications using the HTTP protocol.

Appendix H, *Delayed Responses in HTTP*, presents a method how to modify the request/response communication pattern used in HTTP to mimic the event subscription communication pattern.

Appendix I, *Fundamentals of UPnP*, provides an overview of the fundamentals of the UPnP protocol.

Appendix J, *Data Types in UPnP*, lists common data types used in UPnP.

Appendix K, *Camera Web Interface*, presents a simple web interface publishing pictures taken by the camera.

Appendix L, *Text Encoding on the Web*, discusses text encoding on the web, and possible encoding conflicts.

Appendix M, *Sending Mail with Snapshots*, shows how to send mail including embedded snapshots taken by the camera.

Appendix N, *Tracking Cameras*, shows how the controller application tracks available cameras in the network.

Appendix O, *Certificates and Validation*, gives a short description of how certificates work, and how to install certificates on the Raspberry Pi.

Appendix P, *Chat Interfaces*, shows how to add a chat interface to your devices, making it possible to chat with them using standard chat applications based on XMPP.

Appendix Q, *QR Code*, shows a simple way to generate and display QR code.

Appendix R, *Bill of Materials*, contains bills of materials containing the components used for the boards used in the examples in this book.

These Appendices are not present in the book but are available for download at the following link: `https://www.packtpub.com/sites/default/files/downloads/3494_3532OT_Appendices.pdf`

What you need for this book

Apart from a computer running Windows, Linux, or Mac OS, you will need four or five Raspberry Pi model B credit-card-sized computers, with SD cards containing the Raspbian operating system installed. *Appendix R*, *Bill of Materials*, which is available online, lists the components used to build the circuits used in the examples presented in the book.

The software used in this book is freely available on the Internet:

- A development environment for C#. This can be Xamarin, MonoDevelop, or VisualStudio. The first two are freely available, and the third has a free trial version that can be used.
    - Xamarin, available on Windows and Mac OS, can be downloaded from `http://xamarin.com/download`.

    - MonoDevelop, for Linux (Debian, Ubuntu, Fedora, Red Hat, and openSUSE) can be downloaded from `http://www.monodevelop.com/download/`.
    - A trial version of Visual Studio for Windows can be downloaded from `http://www.visualstudio.com/downloads/download-visual-studio-vs`.

- In *Chapter 4*, *The CoAP Protocol*, we will use the Copper (Cu) Firefox plugin to experiment with CoAP calls. It can be freely downloaded from `https://addons.mozilla.org/en-US/firefox/addon/copper-270430/`.
- For *Chapter 7*, *Using an IoT Service Platform*, and *Chapter 8*, *Creating Protocol Gateways*, we will use the Clayster IoT service platform called `ClaysterSmall`. Free licenses for private, test, or academic use, as well as the Clayster Management Tool, are available for download from `http://www.clayster.com/downloads/`.
- The source code for all the projects presented in this book is available for download from GitHub. See the section about downloading example code, which will follow, for details.

Who this book is for

This book is for developers or electronics engineers who are curious about IoT. With only a rudimentary understanding of electronics (high-school level), Raspberry Pi or similar credit-card-sized computers, and some programming experience using managed code, such as C# or Java, or object-oriented language, such as C++, you will be taught to develop state-of-the-art solutions for the IoT in an instant.

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: "For instance, Digital outputs are handled using the `DigitalOutput` class."