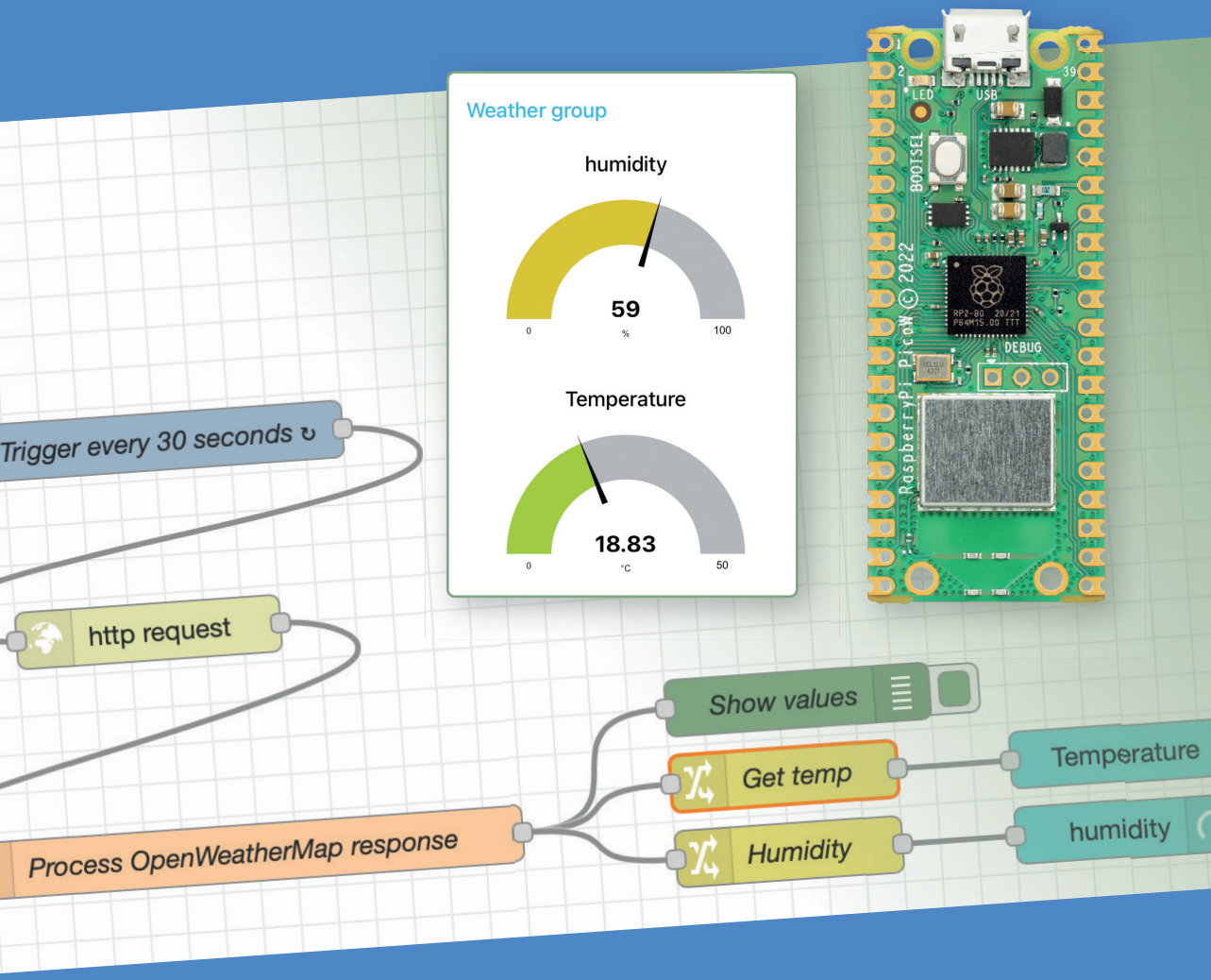
elektor books

Node-RED and Raspberry Pi Pico W

From basics to flows for sensors, automation, motors, MQTT, and cloud services



Dr Peter Dalmaris

elektor
design › share › earn

Node-RED and Raspberry Pi Pico W

From basics to flows for sensors, automation, motors, MQTT, and cloud services

Peter Dalmaris, PhD

elektor
design › share › earn

This is an Elektor Publication. Elektor is the media brand of
Elektor International Media B.V.
PO Box 11, NL-6114-ZG Susteren, The Netherlands
Phone: +31 46 4389444

Declaration

The author, editor, and publisher have used their best efforts in ensuring the correctness of the information contained in this book. They do not assume, and hereby disclaim, any liability to any party for any loss or damage caused by errors or omissions in this book, whether such errors or omissions result from negligence, accident or any other cause. All the programs given in the book are Copyright of the Author and Elektor International Media. These programs may only be used for educational purposes. Written permission from the Author or Elektor must be obtained before any of these programs can be used for commercial purposes.

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

ISBN 978-3-89576-588-9 Print
ISBN 978-3-89576-589-6 eBook

Prepress Production: D-Vision, Julian van den Berg
Printed in the Netherlands

Contents

Did you find an error?

Please let me know.

Go to **txplo.re/nodered**, and fill in the form.

I'll get it fixed right away

About the Author

Dr Peter Dalmaris is an educator, an electrical engineer, electronics hobbyist, and Maker. Creator of online video courses on DIY electronics and author of several technical books. Peter has recently released his book 'Maker Education Revolution', a book about how Making is changing the way we learn and teach in the 21st century.

As a Chief Tech Explorer since 2013 at Tech Explorations, the company he founded in Sydney, Australia, Peter's mission is to explore technology and help educate the world.

Tech Explorations offers educational courses and Bootcamps for electronics hobbyists, STEM students, and STEM teachers.

A lifelong learner, Peter's core skill lies in explaining difficult concepts through video and text. With over 15 years of tertiary teaching experience, Peter has developed a simple yet comprehensive style in teaching that students from all around the world appreciate.

His passion for technology and the world of DIY open-source hardware, has been a dominant driver that has guided his personal development and his work through Tech Explorations.

About Tech Explorations

Tech Explorations creates educational products for students and hobbyists of electronics who rather utilize their time making awesome gadgets instead of searching endlessly through blog posts and Youtube videos.

We deliver high-quality instructional videos and books through our online learning platform, txplore.com.

Supporting our students through their learning journey is our priority, and we do this through our dedicated online community and course forums.

Founded in 2013 by Peter Dalmaris, Tech Explorations was created after Peter realised how difficult it was to find high-quality definitive guides for the Arduino, written or produced by creators who responded to their reader questions.

Peter was frustrated having to search for YouTube videos and blog articles that almost never seemed to be made for the purpose of conveying knowledge. He decided to create Teach Explorations so that he could produce the educational content that he wished he could find back then.

Tech Explorations courses are designed to be comprehensive, definitive and practical. Whether it is through video, e-book, blog or email, our delivery is personal and conversational. It is like having a friend showing you something neat... the "AHA" moments just flow!

Peter left his career in Academia after his passion for electronics and making was rekindled with the arrival of his first Arduino. Although he was an electronics hobbyist from a young age, something the led him to study electrical and electronics engineering in University, the Arduino signalled a revolution in the way that electronics is taught and learned. Peter decided to be a part of this revolution and has never looked back.

We know that even today, with all the information of the world at your fingertips, thanks to Google, and all the components of the world one click away, thanks to eBay, the life of the electronics hobbyist is not easy. Busy lifestyles leave little time for your hobby, and you want this time to count.

We want to help you to enjoy your hobby. We want you to enjoy learning amazing practical things that you can use to make your own awesome gadgets. Electronics is a rewarding hobby. Science, engineering, mathematics, art, and curiosity all converge in a tiny circuit with a handful of components. We want to help you take this journey without delays and frustrations.

Our courses have been used by over 70,000 people across the world. From prototyping electronics with the Arduino to learning full-stack development with the Raspberry Pi or designing professional-looking printed circuit boards for their awesome gadgets, our students

enjoyed taking our courses and improved their making skills dramatically. Here's what some of them had to say:

"*I'm about halfway through this course and I am learning so much. Peter is an outstanding instructor. I recommend this course if you really want to learn about the versatility of the amazing Raspberry Pi*" — Scott

"*The objectives of this course are uniquely defined and very useful. The instructor explains the material very clearly.*" — Huan

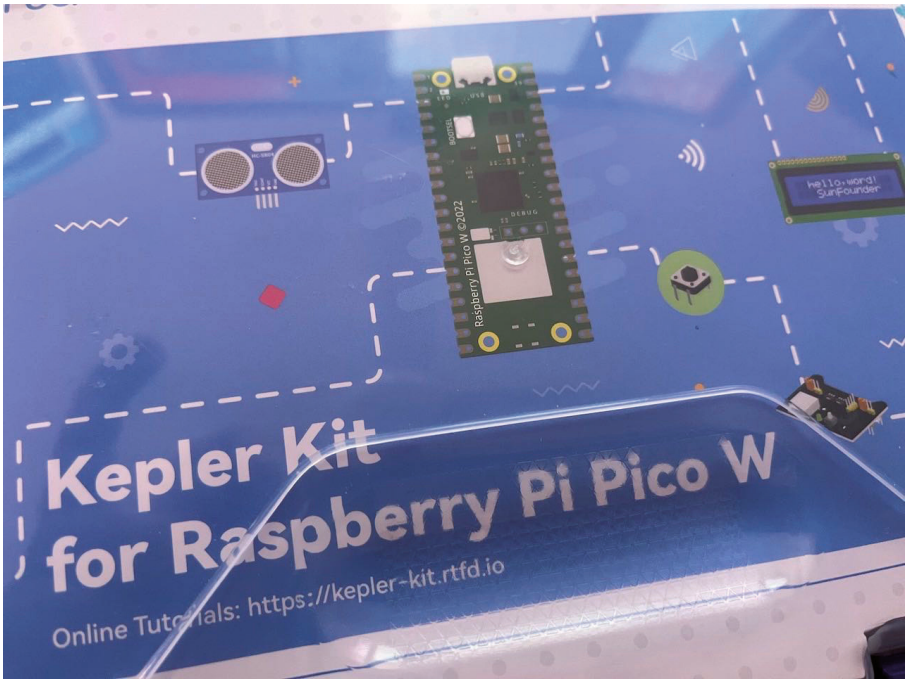
"*Logical for the beginner. Many things that I did not know so far about Arduino but easy to understand. Also the voice is easy to understand which is unlike many courses about microcontrollers that I have STARTED in the past. Thanks*" — Anthony

Please check out our courses at techexplorations.com and let us be part of your tech adventures.

Requirements

You will need a few things to make the most of this book. You probably already have most of all of them:

- A computer running Windows, Mac OS or Linux. If you have a spare Raspberry Pi Model B (any generation) or an old PC, you can use it as your Node-RED server.
- Access to the Internet.
- Hardware (all available in the Kepler kit from Sunfounder: https://txplo.re/kepler):
  - A Raspberry Pi Pico W
  - A breadboard and wires
  - Eight LEDs
  - Resistors (various Ω ratings)
  - Buttons and slide switches
  - Sensors: DHT11, HCSR04, motion PIR sensor, water level sensor, thermistor, analog light sensor.
  - A joystick
  - A 5 V relay
  - An RFID receiver and tag.
  - An infrared receiver and remote control.
  - A 2×16 LCD with the I²C backpack.
  - A 74HC595N THT integrated circuit.
  - A WS2812 RGB LED strip with eight RGB LEDs
  - A small servo motor.
  - A small DC motor



The Kepler kit from Sunfounder contains all the parts you will need for the projects in this book

.

The Book Errata Reporting and Resources Web Page

This book has a web page. This page contains an errata form to report bugs and access related resources as they become available.

Follow this URL to reach the book page: https://txplo.re/nodered.

Foreword

This book is a learning guide and a reference. You can use it to learn Node-RED, Raspberry Pi Pico W, and MicroPython. You can also use it as a source of information for these topics. I have organised this book into four parts to fulfil these dual roles.

Part 1 is dedicated to Node-RED for the absolute beginner. In Part 1, you will learn about Node-RED and event-driven systems, how to install an instance using the Docker option, the basics of nodes and flows, the dashboard and MQTT. If you are new to Node-RED, read the chapters in Part 1 carefully and complete the various projects.

Part 2 brings the Raspberry Pi Pico W into the mix. In the chapters of Part 2, you will learn how to use the Pico W as a Node-RED peripheral. You will learn to use MQTT to enable communications between the Pico and the Node-RED instance. You will also learn how to connect different hardware components to the Pico to implement simple circuits and use Node-RED (and its Dashboard) to control these components or view the data they produce. In Part 2, you will encounter motion, distance and water level sensors, motors, displays, relays, and joysticks, to mention a few. In all projects in Part 2, you will create Node-RED flows and write Raspberry Pi Pico W MicroPython scripts that typically implement an event-driven system.

Part 3 provides a primer to MicroPython. MicroPython is a language specifically designed for embedded systems using Python 3 syntax. Python 3 is one of the most successful programming languages ever. Python's syntax is straightforward, making it easy for beginners to learn. This simplicity encourages good programming practices and allows for a focus on problem-solving rather than syntax issues. MicroPython brings those attributes to Microcontroller programming. If you are not familiar with Python or MicroPython, the chapters in Part 3 will help you learn everything you need to be able to confidently write MicroPython programs for the Raspberry Pi Pico (and Pico W), as well as any other of the many Microcontroller boards that support MicroPython.

Finally, Part 4 provides additional Node-RED resources. These resources consist of content on important Nodes (all explained with the help of mini-projects), control structures, and ways to integrate your Node-RED flows with external services and APIs. In Part 4, you will learn how to create power flows that can be used in more advanced automation settings.

If you are new to Node-RED, Raspberry Pi Pico and MicroPython, I recommend reading this book in a linear fashion. Don't skip anything!

If you are familiar with Node-RED, you can quickly read Part 1 and continue with the projects in Part 2.

If you are unfamiliar with MicroPython, start with Part 3 and continue with other parts of the book. The Part 3 primer on MicroPython is a mini-book in this book and can be used independently of the rest of the content.

In this book, I chose Docker as the infrastructure technology on which I installed my Node-RED instance. You may prefer a different method (and there are several). You are free to choose any installation method you prefer. As long as, in the end, you have an accessible instance of Node-RED and MQTT broker running, you will be able to complete all of the projects in this book.

Enjoy!

Part 1: Node-RED Getting Started

Chapter 1.1 • What is Node-RED?

Node-RED is an open-source flow-based development tool that makes it easy to wire together devices, APIs, and online services. Imagine being able to drag and drop blocks on a screen to create a flowchart that does something—like turning on your lights at sunset or sending you an email when a sensor detects movement. That's what Node-RED lets you do, all without requiring you to write extensive code.

What is flow programming?
Flow programming in the context of Node-RED is a way to build applications by connecting different "nodes" in a flowchart-like manner. Each node performs a specific task, like reading from a sensor, performing a calculation, or sending an email. You create a "flow" by dragging and dropping these nodes onto a canvas and connecting them with "wires" to define the order of operations. The result visually represents your application logic, which you can deploy with a single click.

In traditional text-based programming, you write lines of code to define what your application should do. This often involves setting up loops, conditionals, and functions, and it requires a good understanding of the programming language you're using. In contrast, flow programming in Node-RED abstracts away much of this complexity. Instead of writing code, you're essentially drawing your program. This makes it easier to see the big picture, understand the data flow, and spot any potential issues.

Flow programming shines in scenarios where quick prototyping and iteration are essential. Because you can see the entire logic laid out visually, making changes or adding new functionalities is easier. You don't have to sift through lines of code to find the section you need to modify; you can rearrange or add nodes on the canvas.

It's also beneficial for people who may not have a strong background in programming. The drag-and-drop interface and pre-built nodes make it accessible for beginners, allowing them to focus on solving the problem rather than getting bogged down by syntax and language-specific rules.

Flow programming is compelling for IoT applications and automation tasks. When dealing with multiple devices, sensors, and APIs, the visual nature of flow programming makes it easier to manage the complexity. You can quickly see how data moves from your sensors to your logic nodes and output actions, making the development process more intuitive.

Where can you use Node-RED?
Before continuing, looking at some areas of everyday life where Node-RED is being used with great results is helpful. This will help you understand the impact that Node-RED can have on your projects. To keep this section short, I'll touch only on three areas where Node-RED makes a real difference: industry, education, and home automation.

In industrial settings, Node-RED is a game-changer. For instance, in manufacturing, it can be used to automate entire production lines. By connecting to PLCs (Programmable Logic

Controllers) and sensors, Node-RED can control the sequence of machinery operations. It's also valuable for energy management. Companies use Node-RED to monitor and control how much energy is being used in real time, allowing them to optimise consumption and reduce costs. Beyond that, it's used for predictive maintenance by collecting data from various machinery and using it to predict when a machine is likely to fail, thus scheduling timely maintenance.

Educational institutions find Node-RED to be an excellent tool for teaching and research. It's often used to introduce students to the concepts of IoT, networking, and automation in a hands-on manner. Because of its ease of use, students can quickly move from theory to practice. In research settings, Node-RED is a quick prototyping tool, enabling researchers to test their ideas without spending much time on initial setup and coding.

Node-RED offers endless possibilities for the DIY enthusiast or anyone interested in smart homes. You can set up intelligent lighting systems that adjust based on the time of day or occupancy. You can even create a home security system that sends you alerts based on sensor data. The possibilities are limited only by your imagination and the devices you have at hand.

The tech behind Node-RED

Node.js is an open-source, cross-platform JavaScript runtime environment that executes JavaScript code outside a web browser. Initially released in 2009, it has become a foundational technology for server-side development. Node.js allows you to build scalable network applications using JavaScript, a language traditionally used for client-side scripting in web browsers. It is incredibly lightweight and can run on various platforms, from a Raspberry Pi to a full-scale server.

It uses JSON to represent the flows, making it human-readable and machine-friendly. The logic is encapsulated in "nodes," essentially pre-written JavaScript functions you can string together to create your application.

Node.js is commonly used to build web servers and RESTful APIs. Its non-blocking, event-driven architecture makes it well-suited for handling multiple simultaneous connections, making it a popular choice for real-time applications. Applications like chat rooms and collaborative editing tools often use Node.js to manage real-time, bidirectional communication between the server and clients.

Node.js excels in scenarios where data streaming is crucial. For example, it can process files while uploaded, reducing the overall processing time. Node.js is frequently used in microservices architectures due to its ability to handle multiple small tasks concurrently. Companies like Netflix and Uber employ Node.js to manage their complex, distributed systems.

If you plan to use Node.js solely for Node-RED, you don't need to be an expert in Node.js. A basic understanding of JavaScript and how to install Node.js packages would be sufficient. Node-RED abstracts most of the complexities, allowing you to focus on the logic of your flows rather than the underlying code.

If you can only remember four facts about Node.js, these should be:

1. **Non-blocking, event-driven architecture**: Node.js uses a single-threaded, non-blocking event loop, allowing it to handle many connections simultaneously.

2. **NPM (Node Package Manager)**: Node.js has a rich ecosystem of libraries and frameworks available through NPM that can simplify the addition of new functionalities to your projects.

3. **Cross-Platform:** Node.js can run on various operating systems, including Windows, macOS, and Linux, making it highly versatile.

4. **Community support:** Being open-source and popular means that Node.js has a large, active community contributing to its development and a wealth of tutorials and resources.

Node.js is a powerful runtime environment with many applications beyond Node-RED. Its non-blocking architecture and rich ecosystem make it a go-to choice for many projects, from web servers to real-time communication apps. Even if you're only interested in using it for Node-RED, a minimal understanding will suffice, making it accessible for users with varying coding expertise.

And that's all you need to know about this (unless you plan to build Node.js applications).

Node-RED and microcontrollers
One of the most exciting aspects of Node-RED is its ability to interface with microcontrollers like Arduino and ESP8266. You can read sensor data from these devices and send commands back to control actuators, all through Node-RED's intuitive interface. This makes it an excellent tool for anyone looking to get into hardware hacking or create custom IoT devices.

Node-RED is a versatile and powerful tool with applications ranging from industrial automation to education and home DIY projects. Its drag-and-drop interface makes it accessible for people of all skill levels, while its underlying technology ensures it's robust enough for professional use. Whether you're a hobbyist looking to smarten up your home or an engineer aiming to optimise a production line, Node-RED has something to offer.

Chapter 1.2 • Node-RED in IoT and Event-driven Systems

The Internet of Things (IoT) has revolutionised how small devices are used to monitor and interact with their environment, and how they connect to other systems. As IoT networks grow, the need for an efficient, user-friendly, and powerful tool to manage these connected devices becomes apparent. Node-RED is an open-source, flow-based programming tool that simplifies the creation, deployment, and management of IoT applications and event-driven systems.
Node-RED's flexibility and ease of use make it a perfect fit for various IoT use cases. Here are some of the use cases where Node-RED is particularly well-suited.

Device management and control
Node-RED simplifies the process of connecting and managing IoT devices. It supports many communication protocols, such as MQTT, HTTP, WebSocket, and more, making it easy to collect data from sensors, control actuators, and interact with devices in real-time. For example, a Node-RED flow can be created to monitor a sensor's temperature and humidity data, process the data, and control a smart thermostat based on predefined conditions.

Data processing and analytics
IoT applications often require processing and analysing data from multiple sources. Node-RED provides various built-in nodes for data processing, such as Function nodes (for custom JavaScript code), Switch nodes (for conditional routing), and Change nodes (for modifying message properties). For instance, Node-RED can aggregate sensor data from multiple sources, filter out irrelevant information, and perform calculations to derive insights, such as detecting anomalies or predicting equipment failure.

Integration with cloud services and APIs
Node-RED enables seamless integration with popular cloud platforms, such as AWS, Azure, Google Cloud, and IBM Watson, as well as third-party APIs, like Twitter, Telegram, and Slack. This allows developers to build IoT applications that leverage cloud-based services for data storage, analytics, and machine learning.
An example use case involves using Node-RED to send sensor data to a cloud-based database, perform real-time analytics, and trigger notifications or alerts through third-party messaging services.

Node-RED in event-driven systems
Event-driven systems are designed to respond to specific events or changes in state. Node-RED's flow-based programming model and support for various communication protocols make it an excellent choice for building event-driven applications.

Home automation
Node-RED can create a custom home automation system that responds to events like motion detection, temperature changes, or voice commands. Integrating smart home devices and APIs, Node-RED can control lights, HVAC systems, security cameras, and more based on predefined rules and conditions.

Industrial automation and monitoring
In industrial settings, Node-RED can monitor and control equipment based on events or conditions, such as motors, valves, and conveyor belts. This can help optimise production processes, reduce downtime, and ensure equipment safety.
For example, a Node-RED flow can be created to monitor the status of a production line, detect when a machine malfunctions, and automatically shut down the affected equipment or send alerts to maintenance staff for immediate action.

Smart cities and infrastructure
Node-RED can be crucial in developing innovative city applications that respond to real-time events, such as traffic congestion, air quality, or energy consumption. By integrating with various sensors and services, Node-RED can be used to create intelligent systems that optimise urban infrastructure and enhance the quality of life for residents.
For instance, a Node-RED flow can be designed to analyse traffic data from multiple sources, detect congestion or accidents, and dynamically adjust traffic light timings or notify emergency services.

Node-RED's visual programming interface, extensive library of nodes, and support for a wide range of communication protocols make it a powerful tool for developing IoT and event-driven applications. Its flexibility and ease of use enable developers to create custom solutions for various use cases, from home automation and industrial monitoring to smart cities and infrastructure management.

As the IoT landscape continues to evolve, Node-RED is poised to play a significant role in developing and deploying connected systems that enhance our daily lives and drive innovation across industries

Chapter 1.3 • Communication in Node-RED: Protocols and Methods

Node-RED has an impressive ability to communicate with other systems thanks to its compatibility with various communication protocols. Its flexibility in this domain is key to its success. Let's delve deeper into the protocols and communication methods used by Node-RED.

HTTP and HTTPS

HTTP (HyperText Transfer Protocol) and its secure variant, HTTPS, are fundamental to the web as we know it. They serve as the basis for data communication on the World Wide Web. In Node-RED, you can use the "http" node to make HTTP requests to APIs on the web. For example, you could use an HTTP GET request to fetch data from a weather API or an HTTP POST request to send data to a database.

When working with Node-RED, using HTTP and HTTPS is one of the primary ways to communicate with clients. HTTP is easy to set up, and you don't need special certificates. It's great for quick tests and prototypes. HTTPS, on the other hand, adds a layer of security by encrypting the data. This is crucial if you're dealing with sensitive information.
However, with HTTP (notice the lack of the "S" from the acronym), your data is not encrypted, so transmitting confidential information is unsafe. Anyone can intercept it. HTTPS solves this problem but is a bit more complicated to set up. You'll need to get a security certificate, and sometimes, there might be compatibility issues with older devices or systems.
Later in this book, you will learn how to encrypt communications between your Node-RED server and the web clients using HTTPS.

MQTT

MQTT (Message Queuing Telemetry Transport) is a lightweight messaging protocol often used in IoT applications where bandwidth and power consumption are concerns. With MQTT, devices publish messages to topics, and other devices subscribe to these topics to receive the messages. Node-RED has built-in MQTT nodes, making integrating with MQTT-based systems straightforwardly. For instance, you could have a Raspberry Pi publish temperature sensor data to an MQTT topic and then use Node-RED to subscribe to this topic and react accordingly.

MQTT is a popular choice for sending and receiving data when using Node-RED. As I mentioned earlier, MQTT is lightweight. It doesn't use much bandwidth, which is great if you work with devices with limited resources. It's also good at handling intermittent connections, so MQTT can still work if your network is unreliable. In addition, MQTT is designed for real-time communication, so Node-RED can get updates as they happen.

On the flip side, MQTT has some limitations. It's not the best choice for large data sets because it's designed for small, frequent messages. Also, while MQTT itself is pretty secure, adding extra layers of security can be a bit complex. You might need to integrate it with other security protocols, which could be a hassle if you're new to this.

MQTT is great for real-time, lightweight data communication, but it might not be the best fit for every scenario, especially those requiring high security or large data transfers.

In this book, you will learn how to use MQTT to establish communications between Node-RED and the Raspberry Pi Pico W.

WebSockets
WebSockets is a protocol that provides full-duplex communication between a client and server over a long-lived connection. This is especially useful for applications that require real-time data updates. Node-RED provides a "websocket" node for client and server-side communication using WebSockets. This allows you to create interactive, real-time flows.

WebSockets can be a great way to handle data communications using Node-RED. One of the most interesting features of WebSockets is that they allow for two-way communication between the server and the client. This means you can send and receive data at the same time, making your applications more interactive and responsive. WebSockets are also fast because data can flow freely once the connection is established without repeatedly opening and closing connections.

On the other hand, WebSockets come with their own set of challenges. One issue is that they can be more complex to set up than other protocols like HTTP. You must ensure the client and server are configured correctly to handle WebSocket connections. Not all network configurations also support WebSockets, so you might run into issues if you're behind certain types of firewalls or proxies.

TCP and UDP
TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) are two fundamental protocols of the Internet protocol suite. TCP is connection-oriented and ensures the reliable delivery of a stream of bytes, while UDP is simpler, connectionless, and suitable for scenarios where speed is more important than reliability. Node-RED provides TCP and UDP nodes for handling raw TCP and UDP communications, giving you even more flow flexibility.

If you're working with Node-RED, TCP is a reliable way to send your data. It makes sure all your data packets arrive in the correct order. This is great when you can't afford to lose data, like sending a file or updating a database. TCP also handles congestion well, so if your network is busy, it will adjust to ensure your data gets through.

The downside of TCP is that it can be slower than other methods. It takes time to establish a connection, and the checks it performs to ensure data integrity can add delays. So, TCP might not be the best choice for real-time applications where speed is crucial.

UDP, on the other hand, is all about speed. It sends data without worrying too much about whether it arrives or not. This is good for things like streaming video or audio, where you'd rather lose a few packets than have a delay.

The problem with UDP is that it's not reliable. If ensuring that every piece of data arrives is essential, then UDP is risky. It doesn't guarantee delivery or order. As a result, the recipient might get incomplete or jumbled data.

Serial

Node-RED also supports the serial communication protocol, commonly used for interfacing with hardware such as microcontrollers. With the serial node, you can read data from a serial port or write data to it. This is handy when working with devices like Arduino, which often use serial communication.

Using serial communications with Node-RED can be a straightforward way to communicate with hardware devices. It's a tried-and-true method that's been around for a long time, so it's well-supported and reliable. You'll often find it used in industrial settings or for connecting to older equipment.

Serial communications has its limitations. One of the main drawbacks is that it's not ideal for long-distance communications. The signal can degrade over distance, so it's mainly used for connections that are up to a few meters away. Also, it's generally slower than other modern data transfer methods, so it might not be the best choice for quickly transferring large amounts of data. Another thing to consider is that many modern computers don't come with a built-in serial communications port, so you might need an adapter if your Node-RED instance is running on more modern computer hardware.

In this book, as an example, I have set up my Node-RED instance to run inside a virtual machine, which, in turn, runs inside a modern personal computer without a serial port. Therefore, I cannot use this communication method without an adaptor.

On the other hand, if I had used a Raspberry Pi to host the Node-RED server, I would have been able to create flows that use the Pi's serial port to establish serial communication with microcontrollers such as the Raspberry Pi Pico and the Arduino UNO, both of which have support for serial communications.

Modbus

Modbus is a protocol often used in industrial applications for communication between electronic devices. If you're working in an industrial setting with Modbus equipment, you'll be pleased to know that Node-RED can handle this, too, thanks to community-contributed nodes.

Modbus can be a reliable choice for data communications using Node-RED, especially for industrial applications. One of the big advantages of Modbus is that it is widely used in industrial settings, so it's well-tested and reliable. It's also straightforward to set up. You can use it over various connections, like serial or TCP, giving you some flexibility. Modbus is also suitable for systems that require high reliability, as it has built-in error checking.

On the downside, Modbus has some limitations. It's not the fastest protocol out there, so it might not be the best fit if you need to quickly transfer large amounts of data. Also, while

it's great for simple tasks, it's not as well-suited for more complex data structures. It might be limiting if you need to handle complex data types or commands. Another thing to consider is that it doesn't have built-in security features because it's an older protocol. You'll need to add those yourself if security is a concern.

Modbus is reliable and well-suited for industrial applications but may not be the best choice for high-speed or complex data communications.

Node-RED's extensive protocol support makes it an excellent tool for integrating diverse systems. Whether pulling data from a web API, communicating with an IoT device over MQTT, or reading sensor data from a microcontroller over a serial connection, Node-RED has you covered.

Chapter 1.4 • Node-RED Installation Options

To install and set up a Node-RED server, several options are available. Which one you choose depends on variables such as the resources that you have available and your goals. These options include installing Node-RED to your "regular" work computer as you would with any other program, using a Raspberry Pi or other Linux computers, setting up a Virtual Machine (VM), or leveraging Docker. Each choice has unique features and advantages, but as we move forward, we will focus on the benefits of Docker as a preferred choice.

Installing Node-RED on your local computer
Node-RED is very flexible when it comes to operating systems. You can install it on Windows, Mac OS, and various distributions of Linux.

Installing Node-RED locally has several advantages. For one, you have complete control over your environment. You can customise settings, add nodes, and test flows without worrying about external factors like network latency. It's also easier to integrate with other software and hardware on your computer, such as a webcam and microphone. Plus, you don't have to worry about monthly subscription fees or data limits that you might encounter with cloud-based solutions. Another advantage is that you do not need access to another computer with local installation. You already have a computer, and it's the one that you are (probably) working on right now!

However, the local installation has several significant disadvantages. At the top of the list is that your work computer can be a busy and constantly changing environment. If you are like me, your work computer is where you do all your programming, where you install, remove, reconfigure and experiment with software. It has multiple network connections and lots of connected devices. Once in a while, it will run out of disk space, or other issues will pop up. All this can affect the smooth operation of server software like Node-RED, and you may spend too much time troubleshooting self-inflicted problems rather than learning and using Node-RED. Another issue to consider is that since your flows are stored locally, you won't be able to access them from another computer unless you've set up remote access. And let's not forget, if your computer crashes, you risk losing all your work unless you've been diligent about backups.

For these reasons, I do not recommend installing Node-RED on your work computer unless you install it on a dedicated Virtual machine (see below) so that there is robust isolation between the host and the guest OS.

Raspberry Pi and other Linux computers
By deploying Node-RED on a dedicated Linux machine, such as a Raspberry Pi or an old re-purposed PC, you're positioning yourself for a highly efficient and customisable experience. Let's explore the nuances of this approach, from compatible Linux operating systems to the inherent pros and cons.

Node-RED offers compatibility with a range of Linux distributions. While the Raspberry Pi OS remains the go-to choice for Raspberry Pi users, other distributions like Ubuntu and

Fedora are also viable options. Your selection will likely hinge on the specific requirements of your project or your familiarity with a particular Linux distribution.

Linux's open-source nature is among its most compelling advantages, offering an unparalleled customisation level. Additionally, "headless" Linux distributions are generally lightweight, making them well-suited for hardware with limited computational resources, such as a Raspberry Pi. This results in a cost-effective yet powerful solution for running Node-RED. Furthermore, Linux's reputation for stability and security adds more reliability to your setup.

Perhaps you are unfamiliar with the term "headless", so let's take a moment to explain it. The term "headless Linux" refers to a Linux server or system run without a graphical user interface (GUI). In other words, it doesn't have the graphical desktop environment that is common on desktop computers. Instead, all interactions with the system are done through the command line interface (CLI). Running Linux in headless mode is resource-efficient, as it eliminates the need for the system to allocate resources to the GUI. This is particularly beneficial for servers or systems designed to run specific tasks and don't require user interaction via a graphical interface. It's also a typical setup for Linux instances running on cloud platforms or in data centres.

Despite its merits, Linux has challenges, particularly for those less acquainted with its environment. The operating system often necessitates command-line interactions, which can be daunting for newcomers. Moreover, Linux's robustness and stability come at the cost of limited software compatibility, especially for applications designed with Windows or macOS in mind. However, I recommend using your Linux machine as a Node-RED host only and not as a general-purpose computer; hence, the compatibility issue with applications you find on Windows and macOS computers is a non-issue.

Raspberry Pi, a low-cost, credit-card-sized computer, is one of the most popular choices for running a Node-RED server. Its affordability and accessibility make it a compelling choice for hobbyists and enthusiasts just starting with the Internet of Things (IoT) and Node-RED. Other Linux computers, with their robust and open-source nature, also serve as great hosts for Node-RED.

Installing Node-RED on Raspberry Pi or any other Linux system is straightforward. With a few commands in the terminal, your server will be up and running. However, while these systems are great for smaller, personal projects, they might not scale well for larger, more resource-intensive applications.

Virtual Machine

Deploying Node-RED on a Linux Virtual Machine (VM) offers a unique blend of flexibility and isolation. This approach allows you to run Node-RED in a controlled environment, separate from your primary operating system. The virtual machine can run on your local ("work") computer or another computer in your local network or accessible via the Internet.