AXOOM Gate

Configuration Guide

For

123 – Modbus-TCP Protocol Master Plugin

Rev 2018-07-26 (for AXOOM Gate v 4.106.0.00)

July 26, 2018

Table of Contents

[Chapter 1 Introduction to AXOOM Gate Plugins 3](#_Toc510476503)

[1.1 Overview 3](#_Toc510476504)

[1.2 About AXOOM Gate 3](#_Toc510476505)

[1.3 About Plugins 3](#_Toc510476506)

[1.4 Plugin Deployment Patterns 4](#_Toc510476507)

[1.5 The Plugin Dashboard 5](#_Toc510476508)

[Chapter 2 Connecting to a Modbus Slave 6](#_Toc510476509)

[2.1 About the Modbus Plugin 6](#_Toc510476510)

[2.2 Login to AXOOM Gate 6](#_Toc510476511)

[2.3 Accessing the plugin dashboard 8](#_Toc510476512)

[2.4 Adding connections to Modbus TCP devices 8](#_Toc510476513)

[2.5 Accessing other settings groups 10](#_Toc510476514)

[2.6 Device Status Group 11](#_Toc510476515)

[2.7 Connectivity Group 12](#_Toc510476516)

[Chapter 3 Retrieving Modbus Device Data 14](#_Toc510476517)

[3.1 Modbus Tags Group 14](#_Toc510476518)

[3.2 The Field Mapper Settings 14](#_Toc510476519)

[Appendix A: AXOOM Gate Navigation Icons 17](#_Toc510476520)

[Appendix B: Plugins that ship with AXOOM Gate 18](#_Toc510476521)

# Introduction to AXOOM Gate Plugins

## Overview

This document covers configuring the Modbus-TCP Protocol plugin running in AXOOM Gate (and compatible platforms).

## About AXOOM Gate

AXOOM Gate provides remote and mobile access to industrial production equipment in a secure, efficient, extensible manner. In the interest of operational efficiency and reduced complexity, industrial systems are often configured with data security features disabled. With its built-in, IT-friendly data security settings, AXOOM Gate seamlessly links production systems with office and mobile systems without compromising either factory operation or IT data security requirements.

The installation folder for AXOOM Gate in Microsoft Windows is: C:\Program Files (x86)\AXOOM\AXOOM-Gate. The main executable program is AxoomGateService.exe, which can run both as a Windows service (akin to a Linux daemon) as well as a command-line program.

## About Plugins

An AXOOM Gate plugin is a dynamic-link library (DLL) built with the C-Labs™ C-DEngine™ SDK. Plugins must have a filename with a prefix of “CDMy” or “C-DMy” (examples: CDMyCharts.dll and C‑DMyNetwork.dll).

Plugins enable custom features in AXOOM Gate. The many types of plugins include:

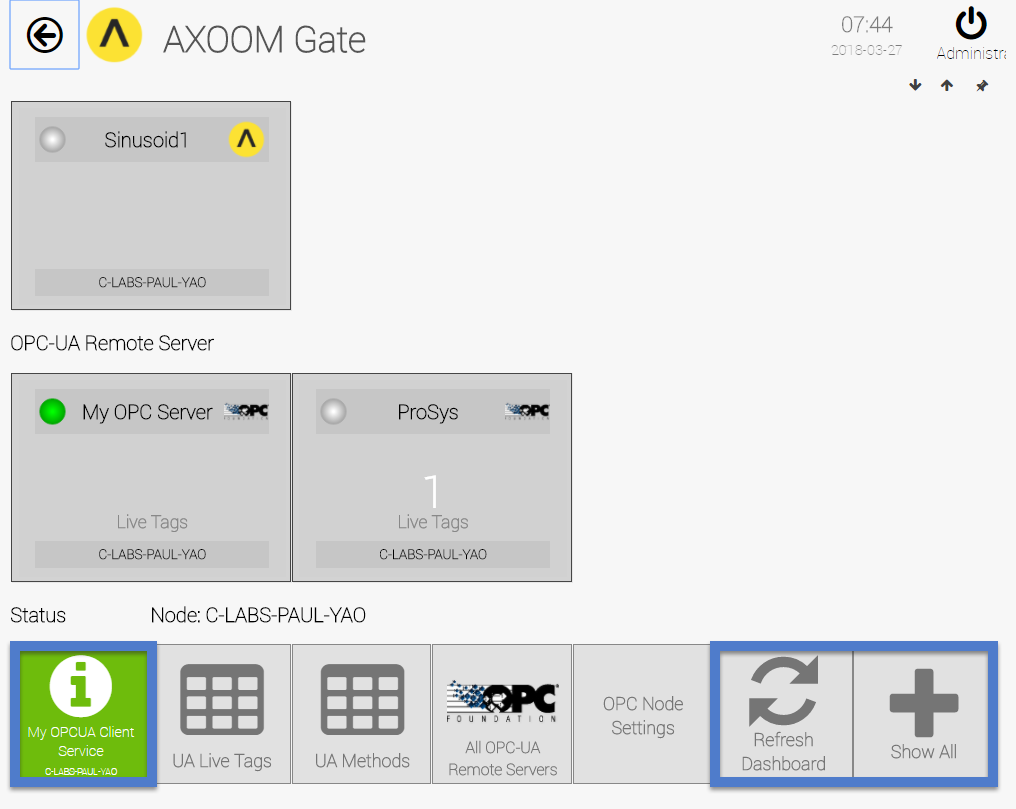
* Configuration Plugins – provide a user interface for configuring hardware or software.
* Connector Plugins – enable a communication channel between AXOOM Gate nodes.
* Device Plugins – enable connections to and data collection from local sensors and devices directly connected to the system running AXOOM Gate.
* User Interface (NMI) Extension Plugins – provide custom controls and other user interface extensions to AXOOM Gate.
* Protocol Plugins – support industry standard protocols like OPC / UA, Modbus, MT Connect, serial ports, and proprietary programmable logic controllers (PLCs) such as the Siemens S7.
* Service Plugins – support other plugins.

As of this writing, there are over 100 plugins. AXOOM Gate v2.102 ships with 22 plugins (see Appendix B for a complete list). Most of these plugins exist to enable live capture of one or more data points from a running system. Such data capture plugins all share a common pattern in terms of how they are configured.

## Plugin Deployment Patterns

To simplify the proper plugin deployment and configuration, this deployment pattern applies to almost every AXOOM Gate plugin:

1. Secure login – A username and password are required to log into AXOOM Gate.
2. Plugin button on home page – All plugins have a button on the AXOOM Gate home page, the first page displayed when you log into AXOOM Gate (see Figure 2.3). Click the plugin’s button to access the plugin dashboard.
3. Plugin dashboard – A plugin dashboard displays all configured items (connections, object, devices, sensors). A plugin dashboard also has a button for creating new items (see Figure 1.1).
4. Editing existing items – Configuring connections involves setting network or device addresses.
5. Once connections are established, you pick specific data points – also known as “properties” – to read.
6. Such properties are grouped together as “Things,” such as you might expect to find in a package that was created to help connect to the “Internet of Things.”

  
**Figure 1.1. A sample plugin dashboard.**

## The Plugin Dashboard

A plugin’s dashboard (see Figure 1.1) provides the primary plugin interface. All “live” items appear at the top of a plugin dashboard. A set of buttons sits below the live items in the plugin dashboard. In the dashboard shown in Figure 1.1, blue rectangles identify the three buttons that are standard on every plugin dashboard:

1) About button: The button with the circled “i” is the About button. Click for details on the dashboard. The button color reflects the status. Status colors, associated status code, and meaning of the status are summarized here:

* + Gray (0): Idle.
  + Green (1): Active / Ok.
  + Yellow (2): Warning.
  + Red (3): Error.
  + Blue (4): Starting / Setup / Ramp Up.
  + Brown (5): Design / Engineering / Configuration.
  + Purple (6): Shutdown.
  + Black (7): Unknown / Unreachable.

2) Refresh Dashboard button: Reloads the dashboard with latest values.

3) Show All button: Open all forms and tables associated with the plugin.

Figure 1.2 shows the default image on a table button. Click a table button to view and edit a table of items managed by the plugin. For example, you can add new items, edit existing item properties, or delete items.

  
**Figure 1.2. Buttons with this table icon enable viewing and editing tables of plugin items.**

# The Modbus TCP Plugin

This chapter covers creating a connection from the Modbus TCP master plugin to a Modbus slave and includes the following topics:

* About the Modbus Plugin
* Login to AXOOM Gate
* Accessing the plugin dashboard
* Adding connections to Modbus devices
* Accessing other settings groups
* The Device Status settings group
* The Connectivity settings group

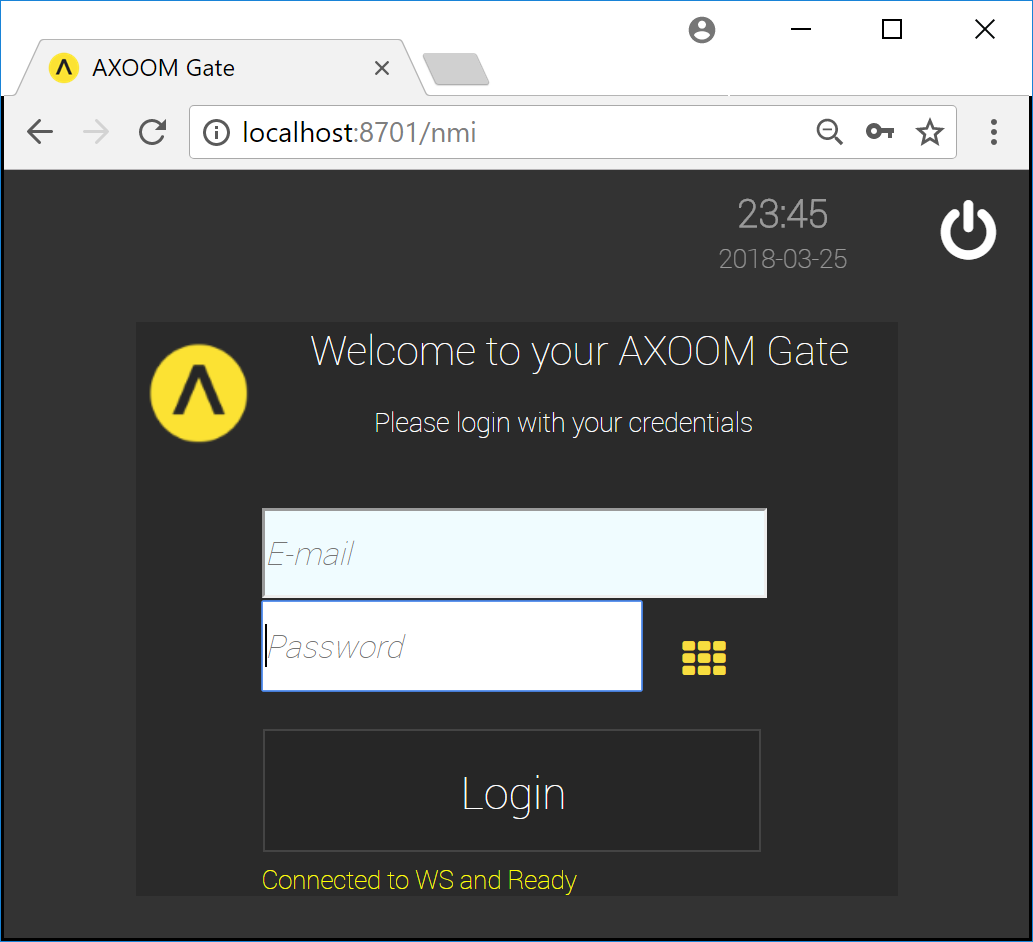
## About the Modbus Plugin

The Modbus protocol is an industry-standard protocol created by Modicon (now part of Schneider Electric) in 1979 to allow serial (RS-232) connections between devices. It is a master-slave protocol, essentially the same as a client-server protocol. Like a server, a slave starts up and listens for requests. A master, like a client, can initiate requests to a slave. The Modbus TCP protocol allows connections to Modbus slaves over a TCP/IP controlled network. This plugin operates as a Modbus master.

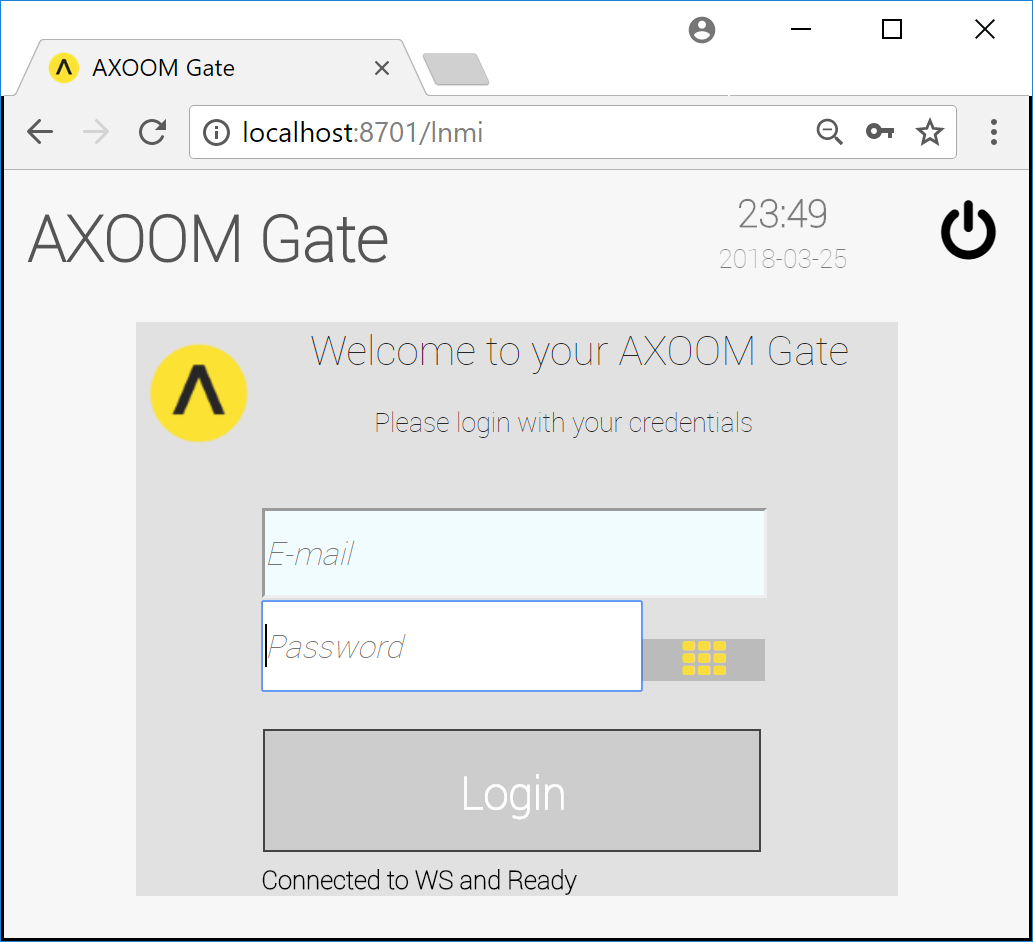
The Modbus protocol is currently overseen by Modbus Organization, Inc, a non-profit industry group. Details about the Modbus Organization are available on their website, [www.modbus.org](http://www.modbus.org). The Modbus specification is available on their website at the following URL: <http://www.modbus.org/specs.php>.

## Login to AXOOM Gate

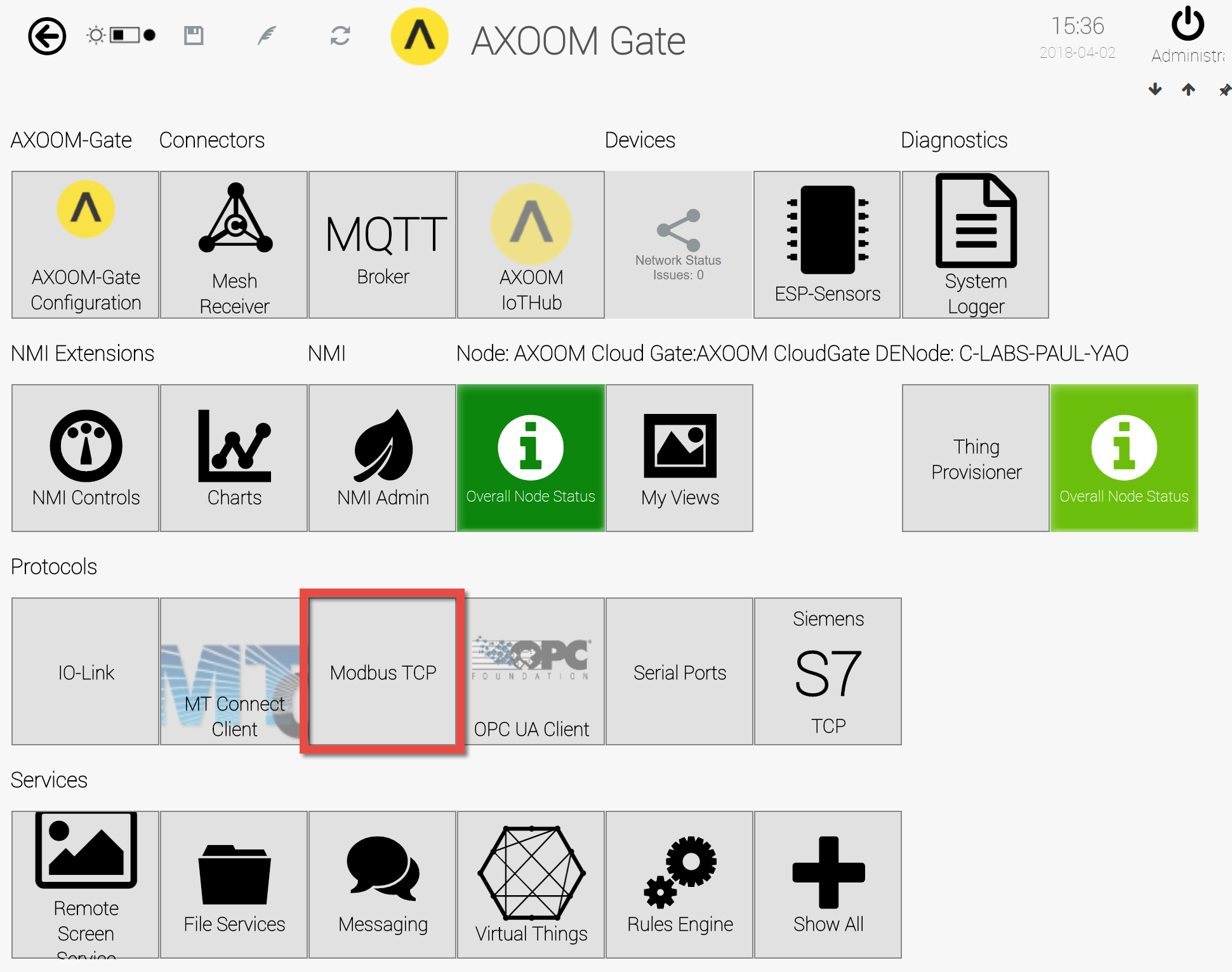
1. Enter this URL, <http://localhost:8701/nmi>, in a web browser. The following login page appears:

  
**Figure 2.1. The dark scheme for the AXOOM Gate login page.**

1. This URL, <http://localhost:8701/lnmi>, enables the “light” display scheme:

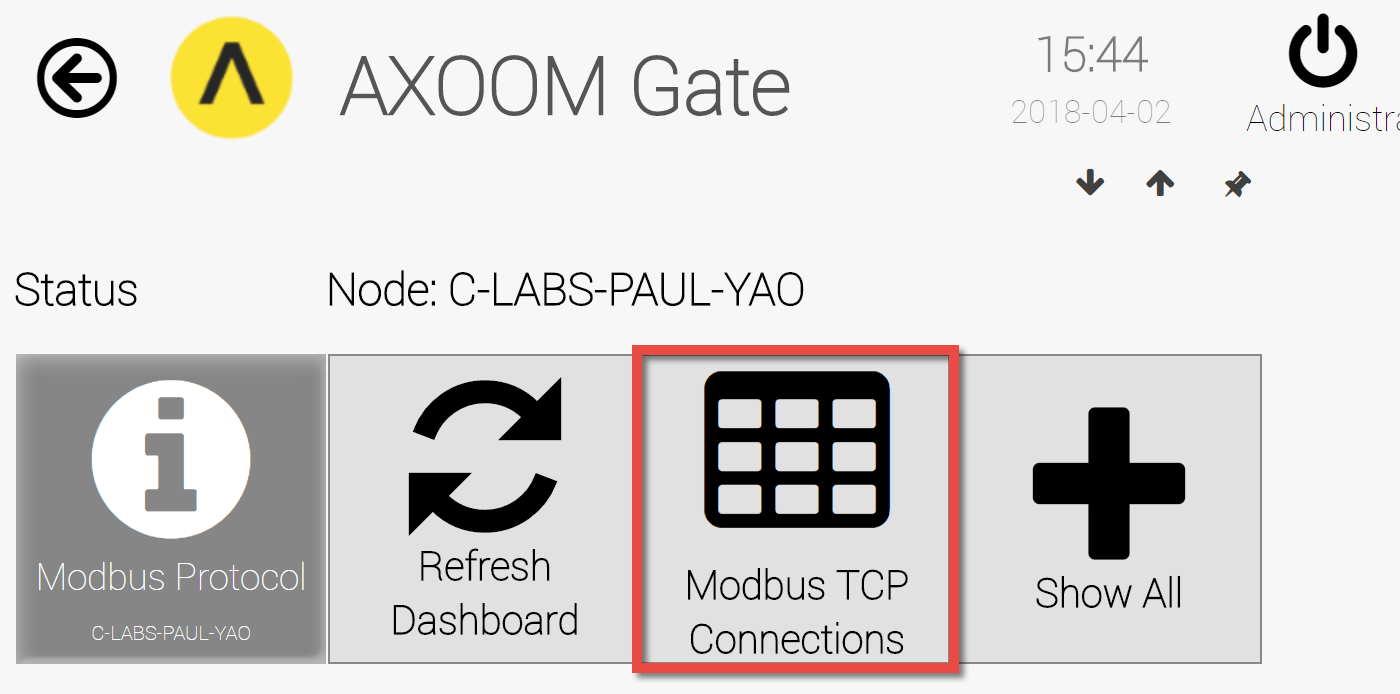
  
**Figure 2.2. The light scheme for the AXOOM Gate login page.**

1. You see the AXOOM Gate home page, as shown in Figure 2.3.

  
**Figure 2.3. The AXOOM Gate home page. The “Modbus TCP” plugin button is highlighted.**

## Accessing the plugin dashboard

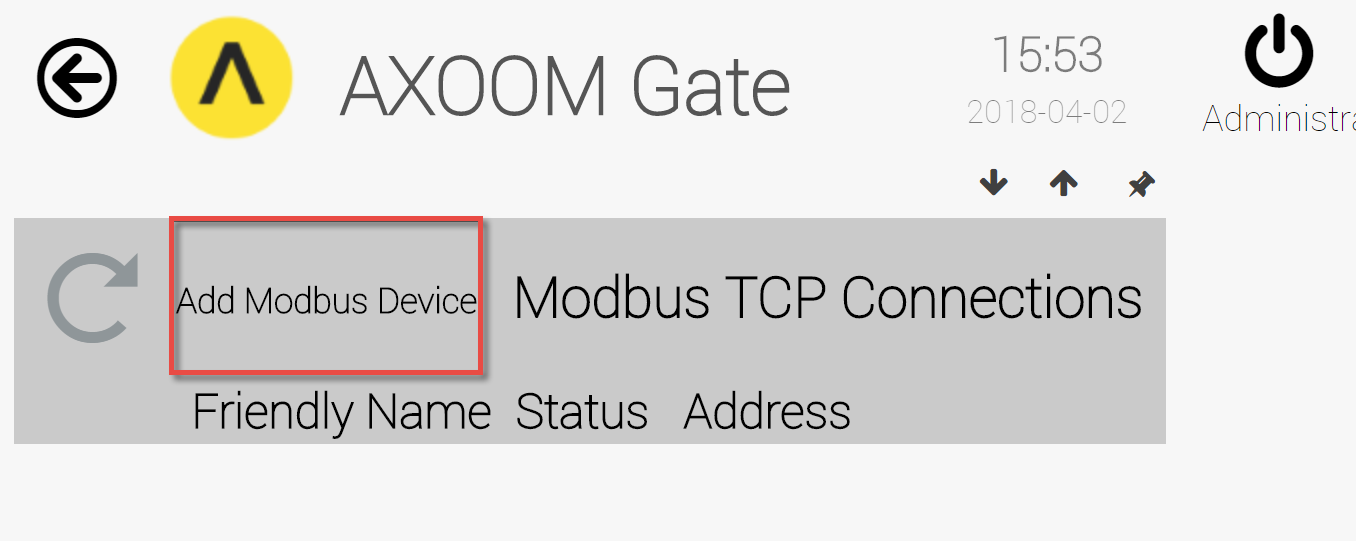
1. When you click the “Modbus TCP” button, the Modbus TCP Master plugin dashboard appears (see Figure 2.4).

**Figure 2.4. The Modbus TCP Master plugin dashboard.**

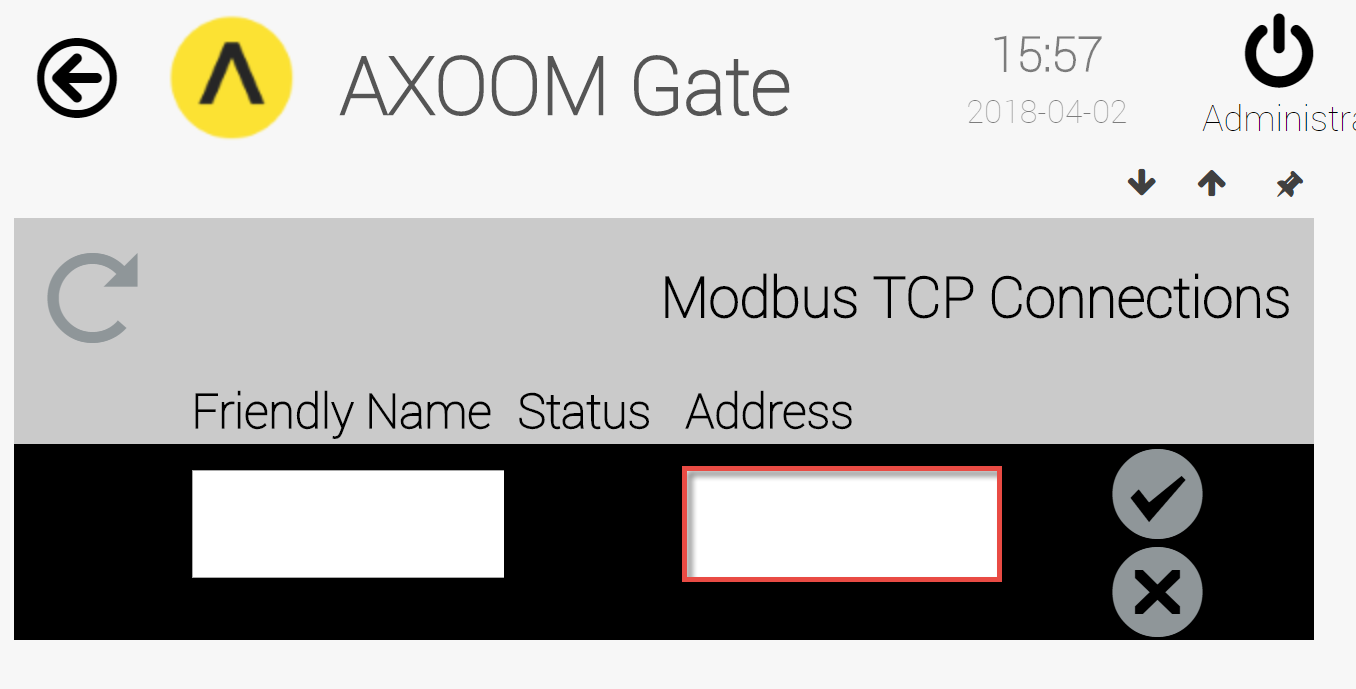
1. To create a connection to a Modbus slave device, click the Modbus TCP Connections button (highlighted).

## Adding connections to Modbus TCP devices

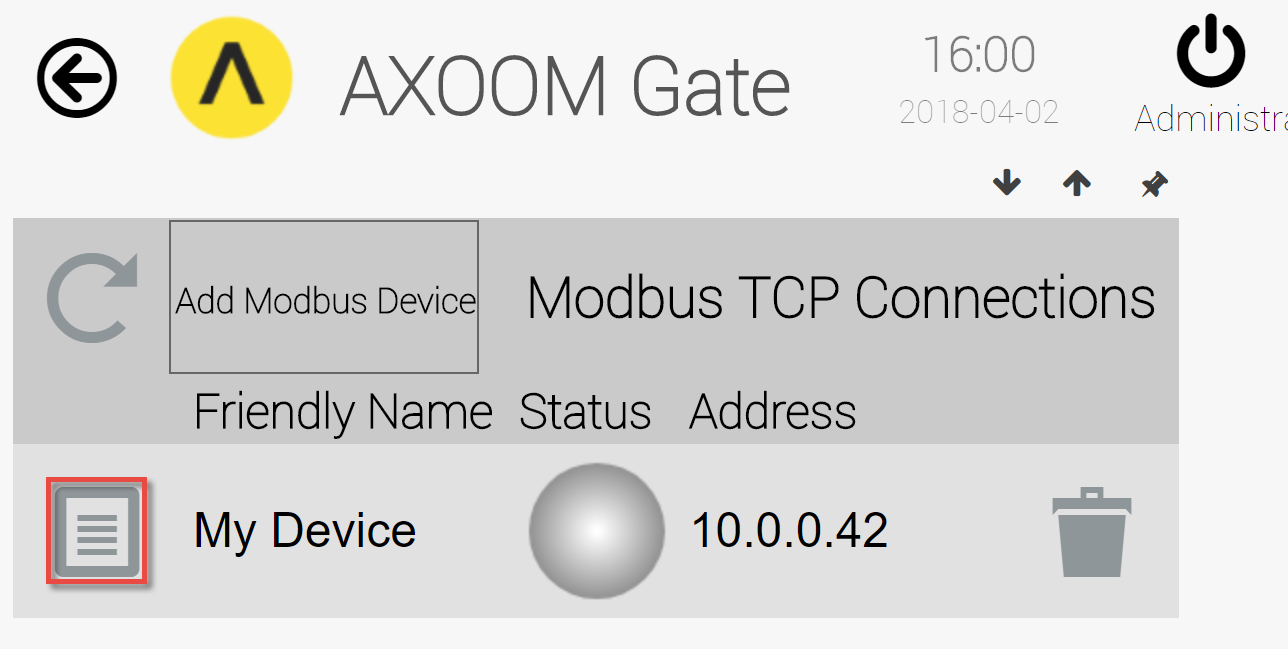
1. Figure 2.5 shows the (now empty) list of Modbus TCP Connections. To connect to a Modbus device, click the "Add Modbus device" button.

  
**Figure 2.5. Click “Add Modbus Device” to create a connection to a device.**

1. An empty row appears, ready for a Modbus TCP device connection details.

  
**Figure 2.6. Enter a Modbus device address. Click checkmark to save.**

1. Enter a value in the “Address” field, then click the checkmark () to save your work. A new connection is added to the list. Note: Do not specify a port in the device address.

  
**Figure 2.7. A connection to a Modbus device.**

The round status light will turn green when a connection is established. It will be red if no connection has been established. Here are more details for each column in the connection list:

| **Column** | **Description** |
| --- | --- |
|  | The Properties icon. Click to view and edit additional settings. |
| Status | Status light. The color will be one of the following:   * + Gray (0): Idle.   + Green (1): Active / Ok.   + Yellow (2): Warning.   + Red (3): Error.   + Blue (4): Starting / Setup / Ramp Up.   + Brown (5): Design / Engineering / Configuration.   + Purple (6): Shutdown.   + Black (7): Unknown / Unreachable. |
| Address | The TCP/IP address for the device. |
|  | Trash can. Delete item on current row.  After deleting, you must refresh the page. |

## Accessing other settings groups

After you create a connection, click the properties button () for more settings.

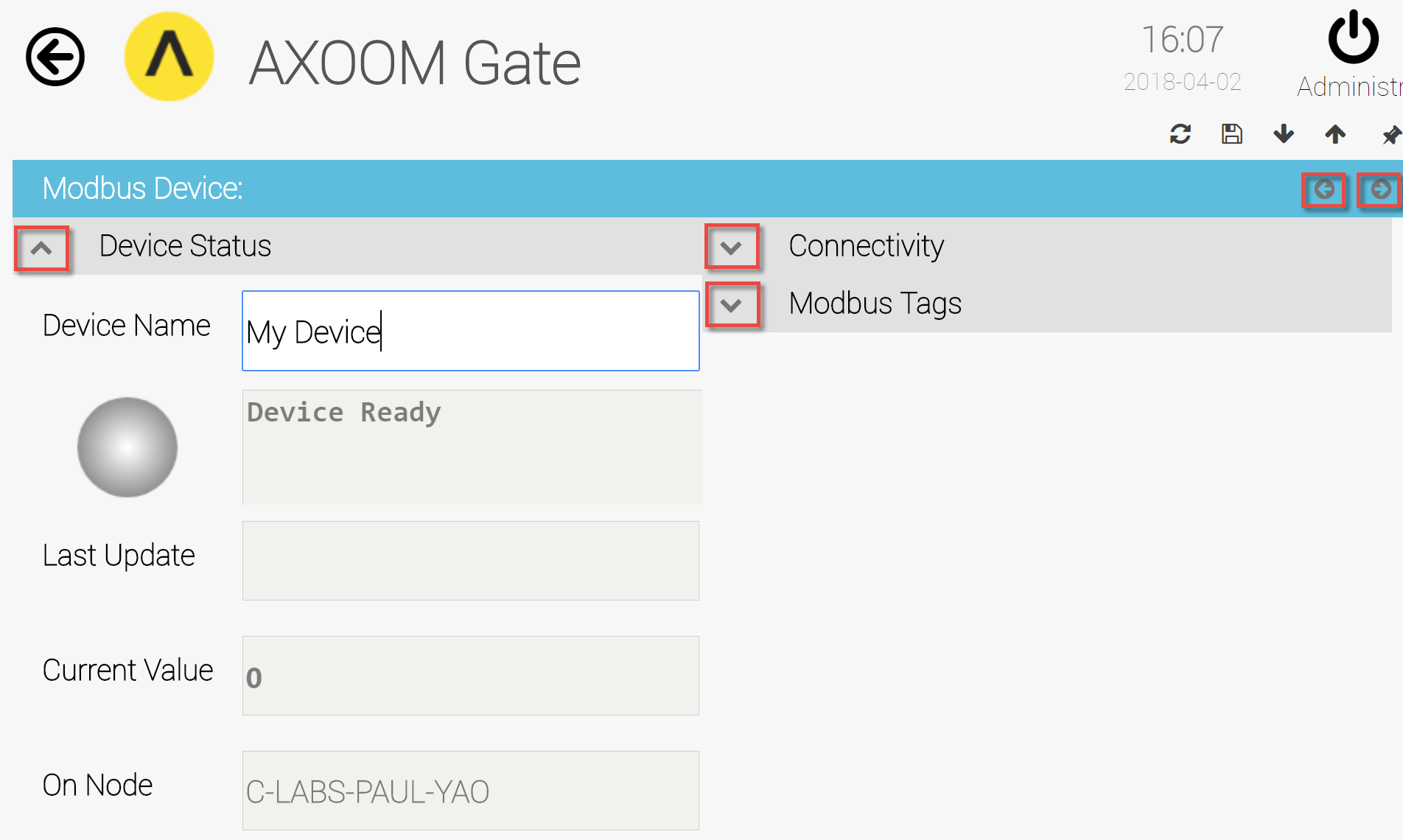
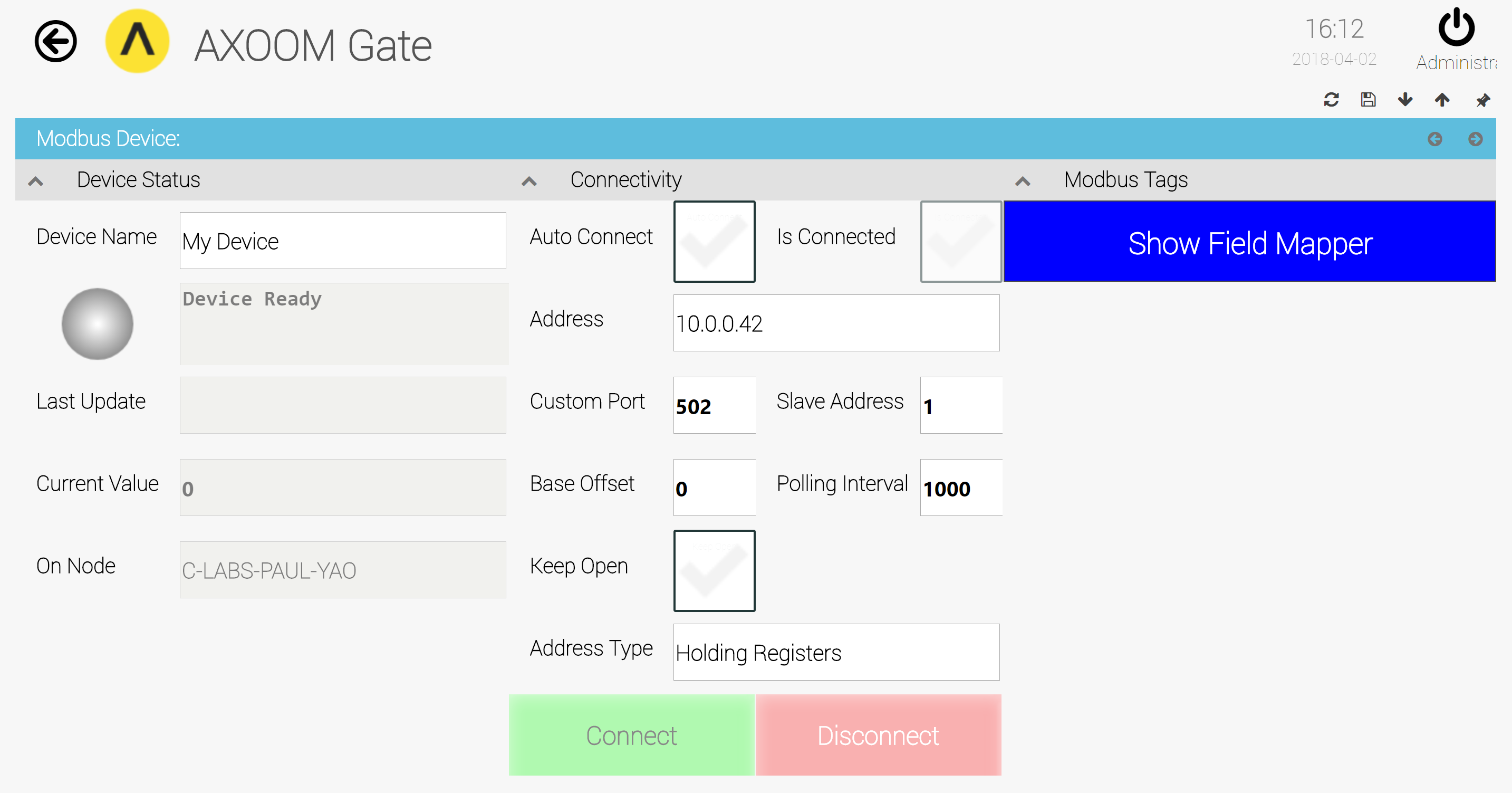
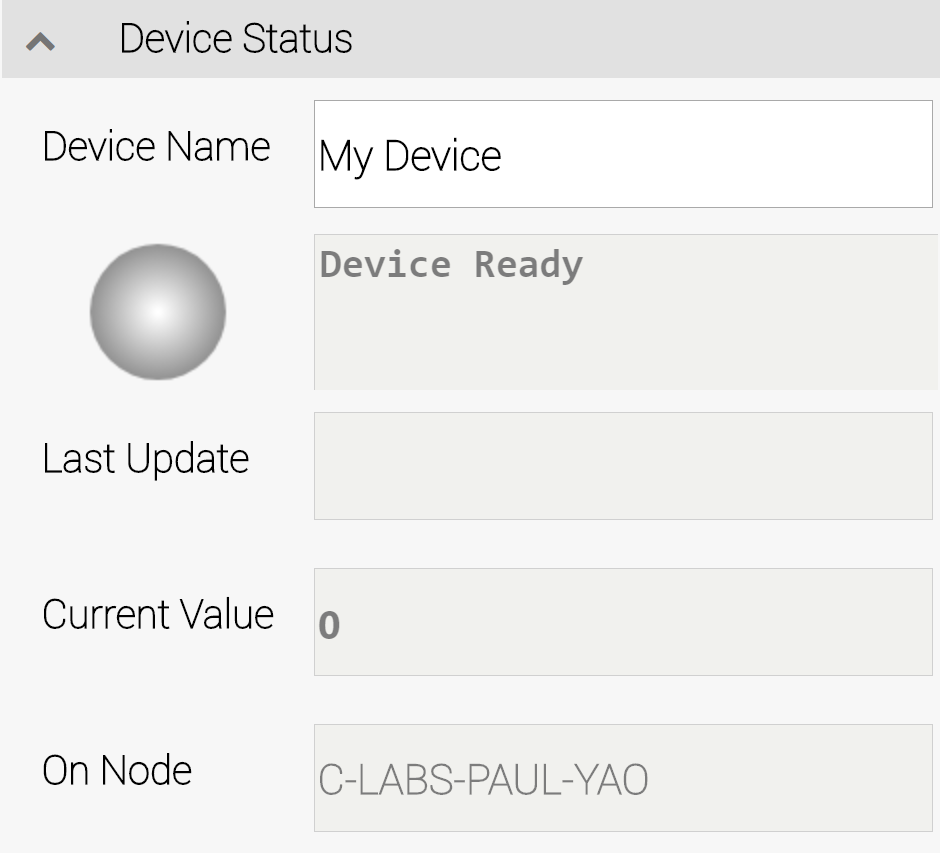
  
**Figure 2.8. When the properties page first opens, one of three settings groups is visible.**

Figure 2.8 shows the property page when it first appears. One group of settings is visible and two groups of settings are hidden. Figure 2.9 shows the property page when all settings groups are visible. This table summarizes the buttons that are available to configure the property settings:

| **Display Buttons** | **Description** |
| --- | --- |
|  | Up arrow. Closes a group. |
|  | Down arrow. Opens a group. |
|  | Left arrow. Decreases the width of browser space used to display property groups. |
|  | Right arrow. Increases the width of browser space used to display property groups. |

 **Figure 2.9. All settings groups are visible on the properties page.**

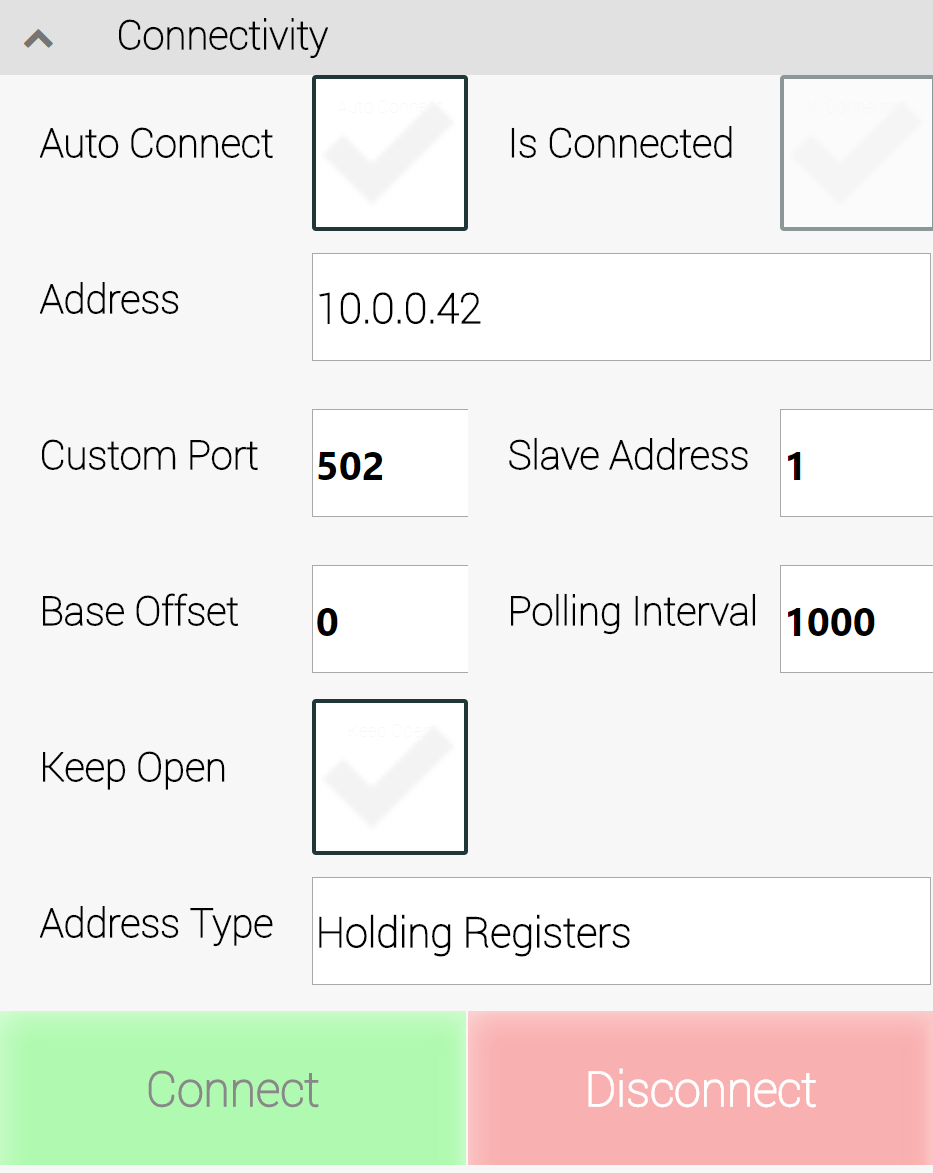
## Device Status Group

 **Figure 2.10. Device Status Group.**

The Device Status Group (see Figure 2.10) holds basic status details, consisting of these fields:

* Device Name – the “Friendly Name.” You can edit this value any time.
* Status light – Green when connected; red when not connected.
* Status text – Status details. When a connection fails, look here for helpful hints.
* Last Update – The client-side timestamp when something was received from the Modbus device.
* Current Value – This value is part of the standard status block and is not used with Modbus.
* On Node – The name of the node where the Modbus slave is running.

## Connectivity Group

  
**Figure 2.11. The Connectivity Group**

You can check whether your server settings are correct, and whether the server is available, by clicking the “Connect**”** button. The indicator labeled “Is Connected” shows the results of your connection attempt. Once connected, click the “Disconnect” button to close a connection.

The **Auto-Connect** setting lets you decide whether to always re-establish a connection when the system starts up (when checked), or whether a user must manually establish the connection.

The Address setting is the network address of your Modbus device. That, along with Custom Port setting, provide the TCP/IP network settings to locate a device.

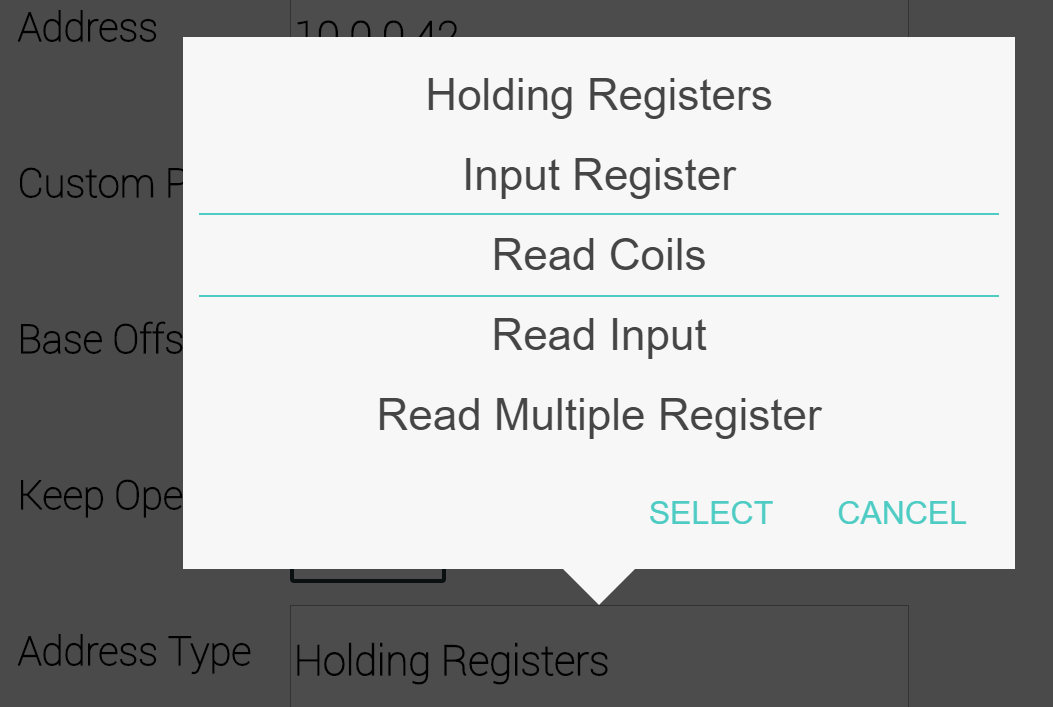
The Slave Address and Base Offset settings

The Polling Interval defines the interval (in milliseconds) to query for values.

The Keep Open field indicates whether a connection should be permanently opened, or whether a connection should be opened, accessed, and then closed.

The Address Type field provides access to list of choices (see Figure 2.12). Available values, as defined in the Modbus specification, including the following:

* Holding Registers
* Input Register
* Read Coils
* Read Input
* Read Multiple Register

 **Figure 2.12. Available Address Type values.**

For details about the different address types please refer to the Modbus Specification and double check with your device what address type is required

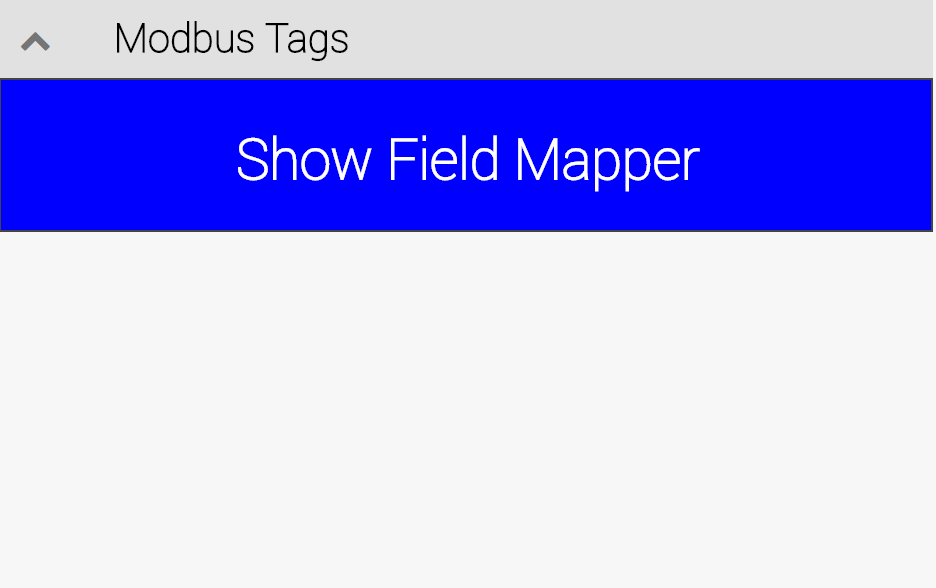
# Retrieving Modbus Device Data

This chapter covers retrieving data from a Modbus device, and includes the following topics:

* Using Configuration Group settings
* Using Tag Import Group settings

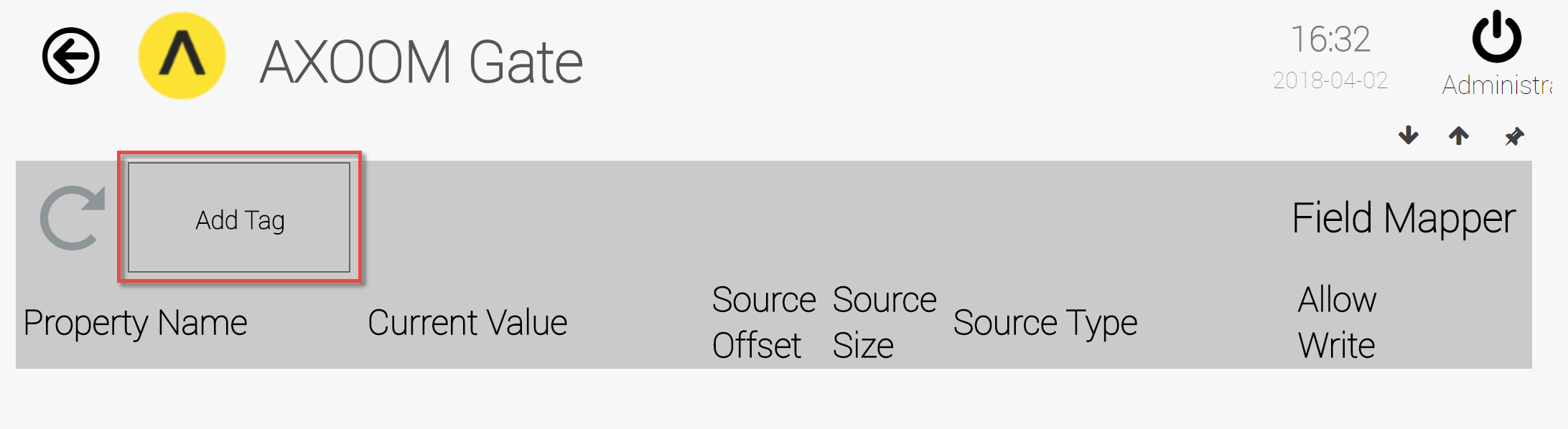
## Modbus Tags Group

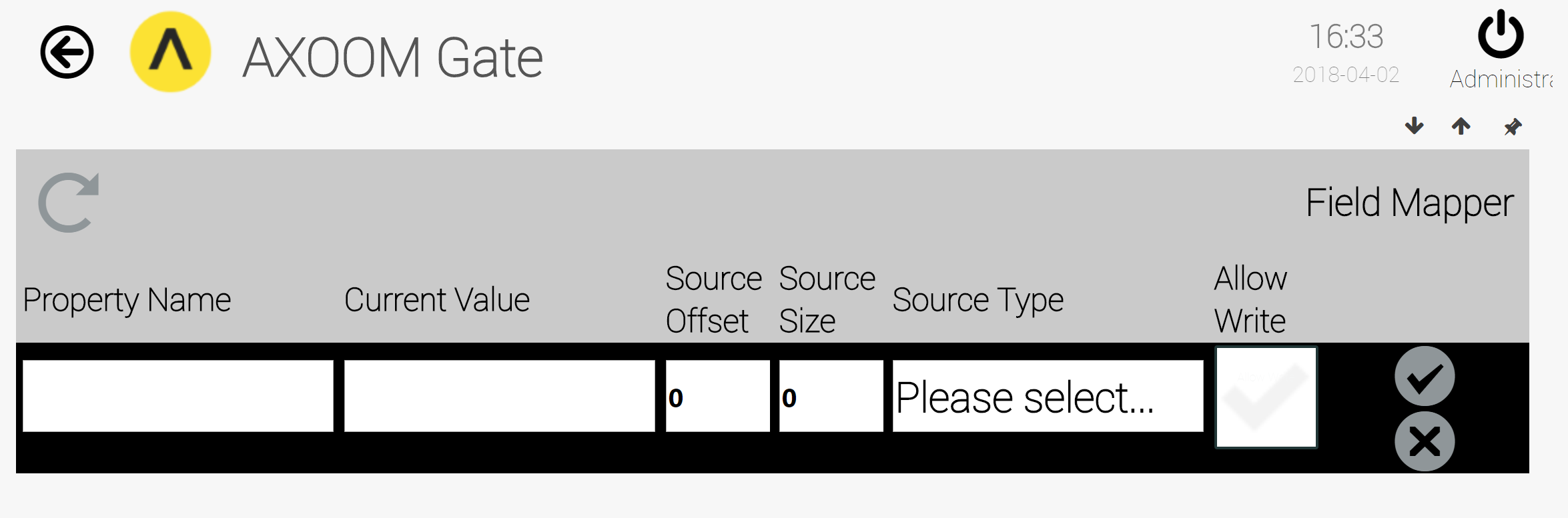
The Modbus Tags Group displays the set of writeable fields that have been configured, along with the current value for each field. As shown in Figure 3.1, there are no currently no writeable fields. Click the Show Field Mapper button to select values to access.

  
**Figure 3.1. The Modbus Tags Group.**

## The Field Mapper Settings

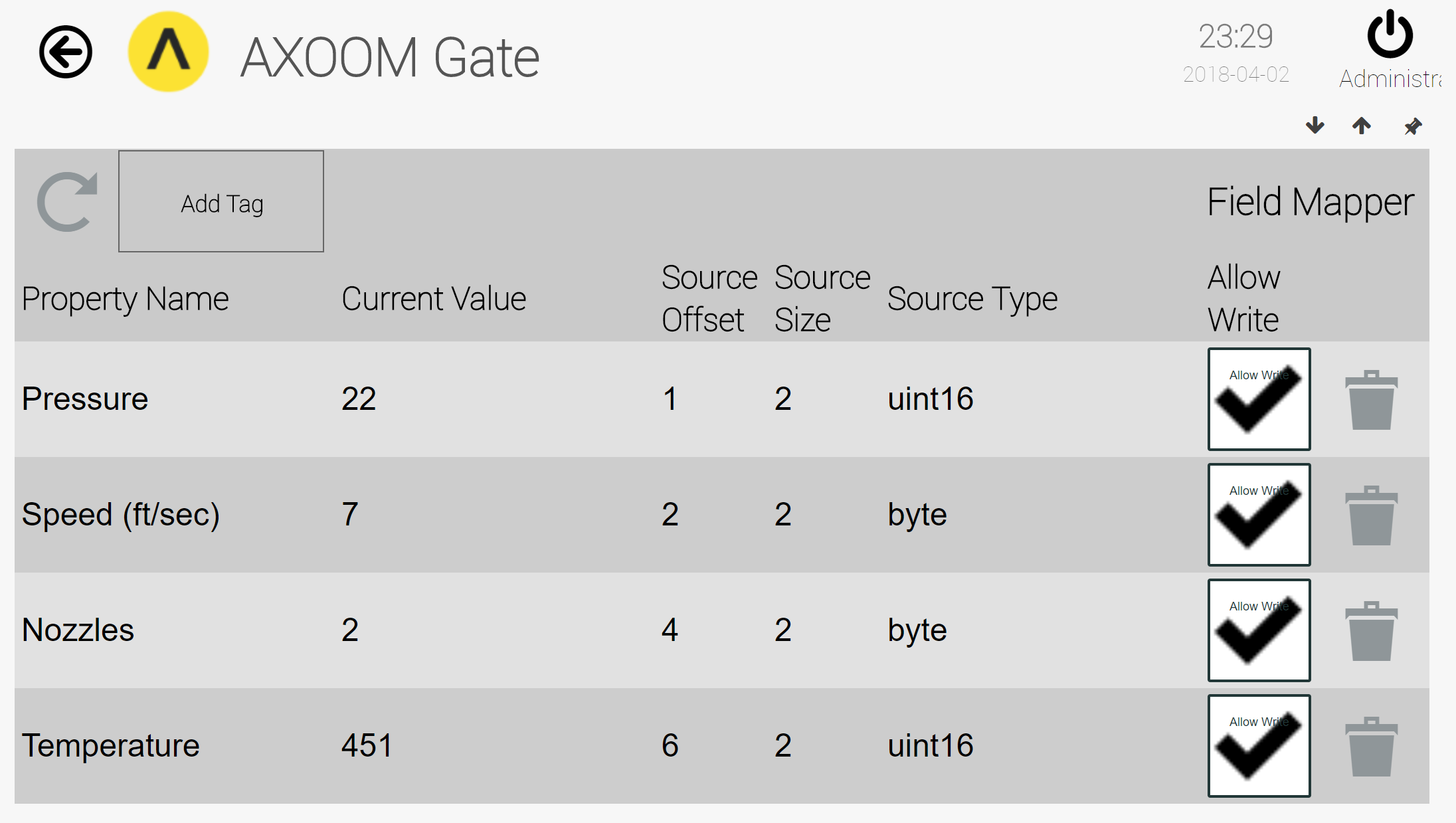
When you click the Show Field Mapper button, you are shown the list of mapped fields, which is initially empty (Figure 3.2). When you click the Add Tag button, a new empty row is displayed, ready for you to enter values (Figure 3.3).

  
**Figure 3.2. The (empty) Field Mapper List.**

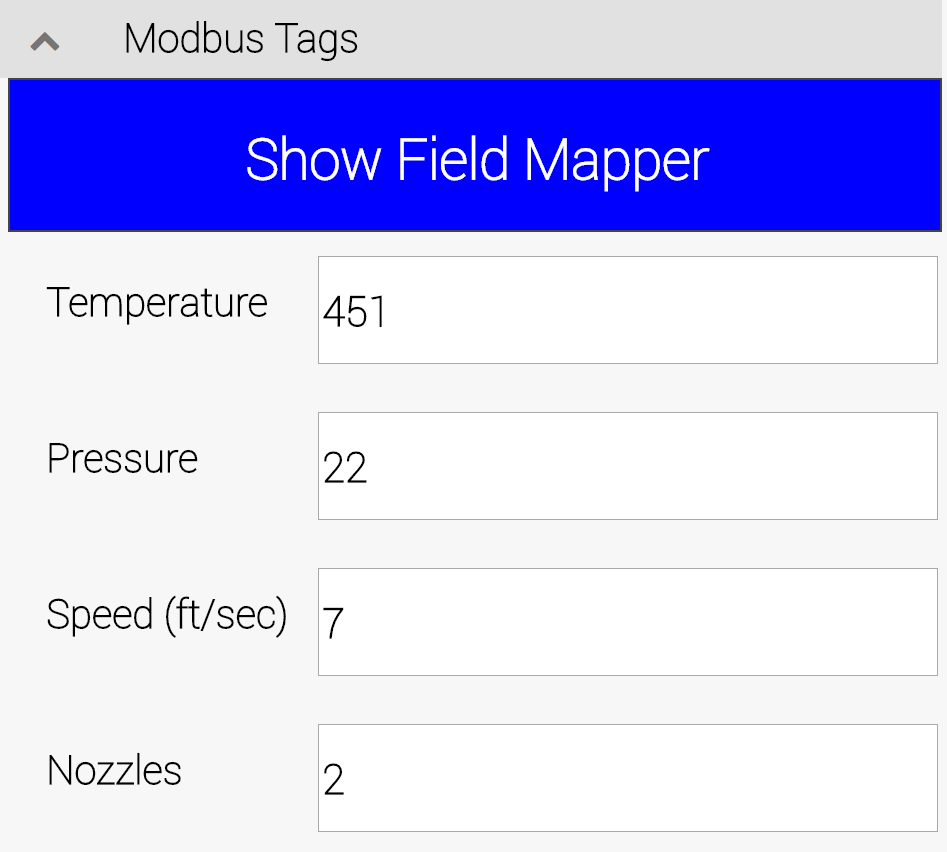
  
**Figure 3.3. A blank row ready to receive field details.**

The field mapper table has six input fields:

* Property Name – This is a “friendly name” to help associate a meaning to a value.
* Current Value – This is a read-only field to display the current value.
* Source Offset – The index to the data value to access.
* Source Size – The number of bytes to include in the data.
* Source Type – This is a data type. You can choose from a list with the following values: byte, double, double-cdab, float, float-abcd, float32, int32, int64, uint16, utf8.
* Allow Write – Whether a field is writeable.

  
**Figure 3.4. Field mapper with four fields defined.**

When the last column (Allow Write) is set to True, the Modbus plugin allows you to edit the value of those fields and the updated values are sent to the Modbus slave. Figure 3.5 shows the four writeable entry fields as defined in Figure 3.4, where the "Allow to Write" flag is set for each of these fields.

  
**Figure 3.5. The entry form for writeable fields.**

# Appendix A: AXOOM Gate Navigation Icons

Here is a summary of the AXOOM Gate navigation icons:

| Icon | Name | Comments |
| --- | --- | --- |
|  | Home | Click to navigate to home page. |
|  | Back | Use instead of the browser’s built-in back button. |
|  | Refresh | Use instead of the browser’s built-in refresh button. |
|  | Properties | Click to view properties. |
|  | Trash Can | Delete an item. |
|  | Up arrow. | Closes a group of controls. |
|  | Down arrow. | Opens a group of controls. |
|  | Left arrow | Decreases the width of browser space used to display property groups. |
|  | Right arrow. | Increases the width of browser space used to display property groups. |

# Appendix B: Plugins that ship with AXOOM Gate

| **File Name** | **Description** |
| --- | --- |
| CDMyAXOOMGate.dll | Manages AXOOM Gate configuration. |
| CDMyAxoomIoTHubSender.dll | Allows for the sending of data to the AXOOM IoT hub. |
| CDMyCharts.dll | Provides charting functionality in a browser. |
| CDMyMeshReceiver.dll | Receives data from a C-DEngine mesh to place in C-DEngine things. |
| C-DMyMessages.dll | A service to send email and other types of messages. |
| DMyModbus.dll | A service for connecting to devices using the Modbus protocol. |
| CDMyMqttBroker.dll | An MQTT broker for receiving thing data from MQTT clients. |
| CDMyMTConnectClient.dll | A protocol transformer plugin to retrieve MT Connect data. |
| C-DMyNetwork.dll | A service for monitoring network services and network devices. |
| CDMyNMIControls.dll | Provides a collection of user interface (NMI) controls. |
| CDMyNMIHtml5.dll | A service that manages the NMI model of the C-DEngine. |
| CDMyOPCUAClient.dll | A protocol plugin to retrieve data from OPC UA tags. |
| CDMyRulesEngine.dll | Rules engine enables setting up of triggers and actions. |
| CDMySerialPort.dll | A service to provide access to data from serial ports. |
| CDMyThingProvisioning.dll | A service that lets you create and configure things. |
| C-DMyVThings.dll | A service to generate data and wrap other (“virtual”) things. |