

# Inhaltsverzeichnis

|                                     |   |
|-------------------------------------|---|
| <b>Konfiguration</b>                | 1 |
| Channel                             | 1 |
| Settings                            | 1 |
| Hinzufügen eines Channels           | 2 |
| Variablen                           | 2 |
| Path                                | 3 |
| Zuordnung von Modbus Function codes | 3 |
| Beispiele                           | 4 |
| Beispiele Strings lesen/schreiben   | 4 |
| <b>Fehlerdiagnose</b>               | 5 |
| Kanal                               | 5 |
| Variablen                           | 6 |
| Logdatei                            | 6 |
| <b>Entities</b>                     | 6 |
| Device                              | 7 |
| Channel                             | 7 |
| Variable                            | 7 |
| <b>Ordner &amp; Dateien</b>         | 7 |
| Ordner                              | 7 |
| Dateien                             | 8 |
| <b>Versionsinformation</b>          | 8 |
| Dieses Dokument                     | 8 |
| Plugin                              | 8 |
| Assembly                            | 8 |



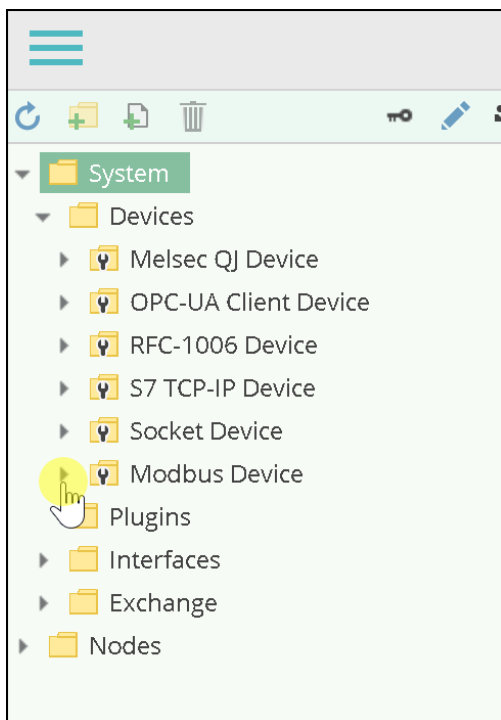
# Modbus Device Plugin

Das Modbus Device Plugin ermöglicht das Lesen und Schreiben von Daten via Modbus TCP.

In der aktuellen Version wird der Betrieb als Modbus TCP Client unterstützt.

## Konfiguration

Die gesamte Modbus Device Plugin-Konfiguration befindet sich unter dem Nodepfad /System/Devices/Modbus Device.



## Channel

Ein Modbus Device Channel repräsentiert die Verbindung zu einem Modbus TCP Server.

## Settings

### IP Address

IP-Adresse des Modbus TCP Servers

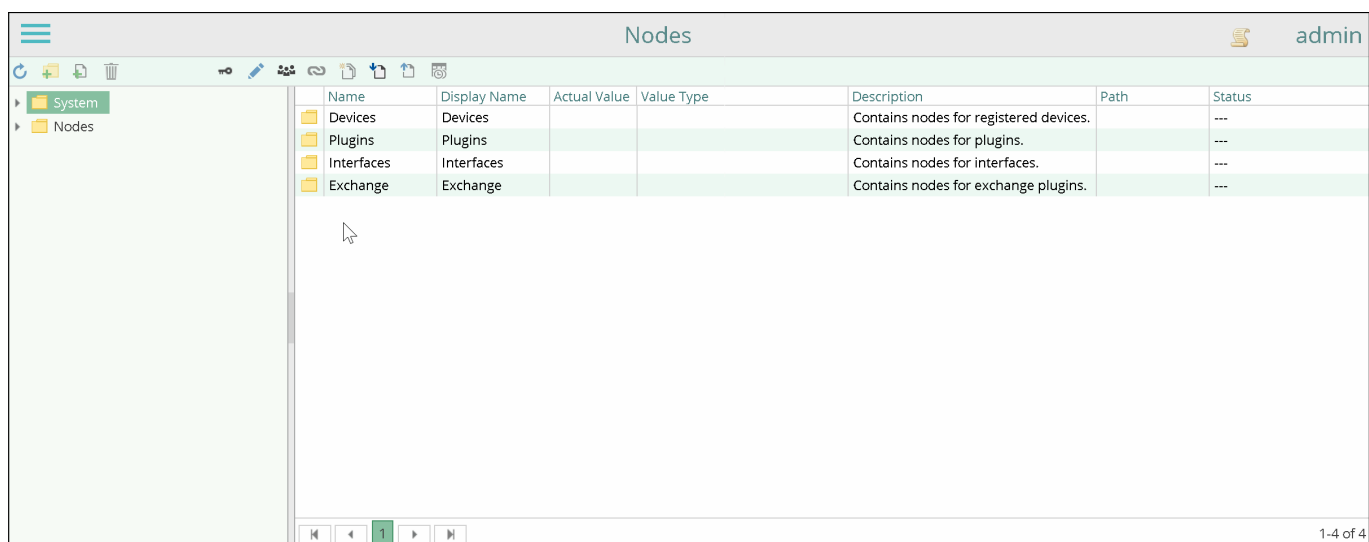
## Port

TCP-Port des Modbus TCP Servers. Der Standard-Port für Modbus ist 502.

## Addressing Mode

Adressierungsmodus für das zu verbindende Gerät. Der Standard ist 1-based. Wenn im Datenblatt des Modbus Geräts die niedrigste Adresse 1 ist, sollte dieser Modus verwendet werden. Beginnt die Adressierung bei 0, sollte der Modus 0-based gesetzt werden.

## Hinzufügen eines Channels



Um einen neuen Modbus-Channel zu erstellen, gehen Sie wie folgt vor:

1. Fügen Sie einen Folder Node unter dem Node Modbus Device/Channels hinzu, oder machen Sie einen Rechtsklick auf den Modbus Device/Channels-Node und wählen Sie Add Channel aus.
2. Tragen Sie im Add Channel-Dialog die Settings für die Modbus TCP Verbindung ein.
3. Nachdem Sie „Save“ geklickt haben, wird die Channel-Node erstellt.
4. Sie können den Kanal starten, indem Sie die Channel-Node auswählen und den Startbutton klicken.

## Variablen

Unter dem Variables-Node können Sie Datenpunktnodes erstellen, die über Modbus gelesen und geschrieben werden.

Die Value Type-Eigenschaft muss dabei auf den entsprechenden Variablentyp festgelegt werden (Boolean, Byte, Int16, UInt16, Int32, UInt32 und die zugehörigen Array-Typen, sowie ab

Version 1.5.11 String).

## Path

Über die Path-Eigenschaft des Nodes wird die Modbus-Adresse, der Datentyp im Modbus Server und die Länge der Daten (bei Arrays) definiert:

```
<UnitID>.<Entity><StartAddress>.<Offset>,<Length>,<DataType>
```

| Platzhalter    | Beschreibung                                                                                                                                  | Mögliche Werte (Bedeutung)                                                                                                                                                                                         | Optional (Standard Wert)                                                                                       |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| <UnitID>       | Unit ID des Modbus Slaves                                                                                                                     | 0 - 255                                                                                                                                                                                                            | Ja (0)                                                                                                         |
| <Entity>       | Objekttyp im Modbus Slave                                                                                                                     | c - Coil<br>di - Discrete Input<br>hr - Holding Register<br>ir - Input Register                                                                                                                                    | Ja (c)                                                                                                         |
| <StartAddress> | Startadresse der Daten                                                                                                                        | 0 - 9999<br>Die führende Ziffer einer Modbusadresse definiert den Objekttypen, auf den zugegriffen wird. Dieser wird durch <Entity> festgelegt, während die führende Ziffer bei der Startadresse weggelassen wird. | Nein                                                                                                           |
| <Offset>       | Bei Zugriff auf Bits in Registern: Bit-Offset im 16-Bit Register<br>Offset in der Einheit des definierten Datentyps (wenn <DataType> != bit ) | 0 - 15 (bei Zugriff auf Bits)<br>Abhängig von den gültigen Adressen im Modbus Server                                                                                                                               | Ja (0)                                                                                                         |
| <DataType>     | Datentyp im Modbus Slave                                                                                                                      | bit = bool<br>byte<br>word = uint16<br>dword = uint32                                                                                                                                                              | Ja<br>(bit wenn <DataArea> == c oder di bzw. ein <Offset> angegeben ist<br>word wenn <DataArea> == hr oder ir) |

## Zuordnung von Modbus Function codes

| Function code | Beschreibung          | Entsprechung                                                                   |
|---------------|-----------------------|--------------------------------------------------------------------------------|
| 0x01          | Lese Coils            | Lese Variabel des Typs Boolean oder Boolean-Array mit Objekttyp Coil           |
| 0x02          | Lese Discrete Inputs  | Lese Variabel des Typs Boolean oder Boolean-Array mit Objekttyp Discrete Input |
| 0x03          | Lese Holding Register | Lese Variabel des Typs UInt16 oder UInt16-Array mit Objekttyp Holding Register |

| Function code | Beschreibung                        | Entsprechung                                                                 |
|---------------|-------------------------------------|------------------------------------------------------------------------------|
| 0x04          | Lese Input Register                 | Lese Variabel des Typs UInt16 oder UInt16-Array mit Objekttyp Input Register |
| 0x05          | Schreibe einzelne Coil              | Schreibe Variabel des Typs Boolean mit Objekttyp Coil                        |
| 0x06          | Schreibe einzelnes Holding Register | Schreibe Variabel des Typs UInt16 mit Objekttyp Holding Register             |
| 0x0F          | Schreibe mehrere Coils              | Schreibe Variabel des Typs Boolean-Array mit Objekttyp Coil                  |
| 0x10          | Schreibe mehrere Holding Register   | Schreibe Variabel des Typs UInt16-Array mit Objekttyp Holding Register       |

## Beispiele

| Value Type | Path                            | Erklärung                                                          |
|------------|---------------------------------|--------------------------------------------------------------------|
| Boolean    | 16 oder 0.c16,1,bit             | Bit an Coil Adresse 16 in Unit 0                                   |
| Boolean    | 1.di0 oder 1.di0,1,bit          | Bit an Discrete Input Adresse 0 in Unit 1                          |
| UInt16     | 1.hr20 oder 1.hr20,1,word       | 16-Bit Wert an Holding Register Adresse 20 in Unit 1               |
| UInt16     | hr20,4 oder 0.hr20,4,word       | Array aus 4 16-Bit Werten an Holding Register Adresse 20 in Unit 0 |
| UInt32     | ir24,uint32 oder 0.ir24,1,dword | 32-Bit Wert an Input Register Adresse 24 in Unit 0                 |
| Bool       | ir24.1 oder 0.ir24.1,1,bit      | Bit an Bitoffset 1 an Input Register Adresse 24 in Unit 0          |

|                                                                                     | Name      | Display Name | Actual Value    | Value Type   | Description | Path           | Status |
|-------------------------------------------------------------------------------------|-----------|--------------|-----------------|--------------|-------------|----------------|--------|
|  | DO0       | DO0          | True            | Boolean      |             | 16             | Good   |
|  | DO1       | DO1          | True            | Boolean      |             | c17            | Good   |
|  | DO2       | DO2          | True            | Boolean      |             | 0.c18          | Good   |
|  | DO3       | DO3          | False           | Boolean      |             | 0.c19,bool     | Good   |
|  | AI0       | AI0          | 4095            | UInt16       |             | ir0            | Good   |
|  | AI1       | AI1          | 4095            | UInt16       |             | 0.ir1          | Good   |
|  | AI2       | AI2          | 4095            | UInt16       |             | 0.ir2,word     | Good   |
|  | AI3       | AI3          | 4095            | UInt16       |             | 0.ir3,uint16   | Good   |
|  | AO0       | AO0          | 123999          | UInt32       |             | hr8,dword      | Good   |
|  | AO1       | AO1          | 123039          | UInt32       |             | 0.hr10,dword   | Good   |
|  | AO2 + AO3 | AO2 + AO3    | [1040596, 1203] | UInt32-Array |             | 0.hr12,2,dword | Good   |

## Beispiele Strings lesen/schreiben

Ein String-Wert kann maximal **255 Zeichen** beinhalten. Als String Encoding wird ISO-8859-1 verwendet.

Wird ein String mit [ ] angegeben, wird der eingegebene Wert als Länge definiert. Die Arraylänge wird dann ignoriert.

Strings können von Holding Register, Coils, Discrete Inputs und Input Registers gelesen werden.

Nur im Holding Register können String-Werte gelesen und geschrieben werden.

| Path               | Lesen | Schreiben | Erklärung                                                                       |
|--------------------|-------|-----------|---------------------------------------------------------------------------------|
| hr01,19,string     | X     | X         | Hier wird im Holding Register ein String von der Länge 19 definiert.            |
| hr01,string        | X     | X         | Hier wird im Holding Register ein String von der maximalen Länge 255 definiert. |
| hr01,21,string[19] | X     | X         | In diesem Beispiel wäre die Länge 19 und nicht 21!                              |
| 1.c1,1,string[20]  | X     |           | Hier wird in Coils ein String von der Länge 20 definiert.                       |
| 1.di,1,string[20]  | X     |           | Hier wird in Discrete Inputs ein String von der Länge 20 definiert.             |
| 1.ir1,1,string[20] | X     |           | Hier wird im Input Register ein String von der Länge 20 definiert.              |

|  | Name                | Display Name        | Actual Value | Value Type | Description | Path               | Status |
|--|---------------------|---------------------|--------------|------------|-------------|--------------------|--------|
|  | testHoldingRegister | testHoldingRegister | test         | String     |             | hr01,1,string[19]  | Good   |
|  | testCoil            | testCoil            | Hallo!       | String     |             | 1.c1,1,string[20]  | Good   |
|  | testDiscreteInput   | testDiscreteInput   | Hallo!       | String     |             | 1.di1,1,string[20] | Good   |
|  | testInputRegister1  | testInputRegister1  | Hallo!       | String     |             | 1.ir1,1,string[20] | Good   |

## Fehlerdiagnose

Das Modbus Device Plugin liefert je nach zu untersuchender Schicht verschiedene Statusinformationen. Generell werden die kanalbasierten Diagnoseinformationen durch den Verbindungsstatus des Channels zum Modbus produziert. Die variablenbasierten Diagnoseinformationen werden während des Lese-/Schreibzugriffs auf die verschiedenen Variablen produziert.

## Kanal

Um den Status des Modbus-Kanals zu überwachen und zu diagnostizieren, werfen Sie einen Blick auf das folgende Bild:

Channel 1

Could not create connection: A connection attempt failed because the connected party did not properly respond aft...

| Name      | Display Name | Actual Value | Value Type | Description                              | P |
|-----------|--------------|--------------|------------|------------------------------------------|---|
| Settings  | Settings     |              |            | Defines the different settings which ... |   |
| Control   | Control      |              |            | Provides device channel control mec...   |   |
| Status    | Status       |              |            | Provides information about the devi...   |   |
| Variables | Variables    |              |            |                                          |   |

Das obige Bild zeigt das Bedienfeld des Modbus-Kanals, das alle statusrelevanten Informationen anzeigt. Das Bedienfeld aktualisiert automatisch seine Statusinformation, wenn ein neuer Status verfügbar ist.

## Statuskreis

| Farbe                                                                             | Bedeutung                                                                                                                    |
|-----------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
|  | Der Kanal ist gestoppt. Klicken Sie den ►-Button, um ihn zu starten.                                                         |
|  | Der Kanal startet oder stoppt gerade oder wartet auf den Verbindungsaufbau.                                                  |
|  | Der Kanal läuft und es wurde erfolgreich eine Verbindung hergestellt. Sie ihn können durch Klick auf den ■-Button stoppen.   |
|  | Der Kanal läuft, aber die Verbindung ist momentan fehlerhaft. Bitte überprüfen Sie den Statustext für weitere Informationen. |

## Variablen

Um den Status der verschiedenen Variablen zu überwachen und zu diagnostizieren, werfen Sie einen Blick auf die in CoDaBix® angezeigte Status-Eigenschaft der Spalte. Benutzen Sie den Button „Read actual Value“, um die Werte über Modbus auszulesen und das Ergebnis in den Variablen zu speichern.

|                      | Name      | Display Name | Actual Value    | Value Type   | Description | Path           | Status |
|----------------------|-----------|--------------|-----------------|--------------|-------------|----------------|--------|
| System               | DO0       | DO0          | True            | Boolean      |             | 16             | Good   |
| Devices              | DO1       | DO1          | True            | Boolean      |             | c17            | Good   |
| Melsec QJ Device     | DO2       | DO2          | True            | Boolean      |             | 0.c18          | Good   |
| OPC-UA Client Device | DO3       | DO3          | False           | Boolean      |             | 0.c19,bool     | Good   |
| RFC-1006 Device      | AI0       | AI0          | 4095            | UInt16       |             | ir0            | Good   |
| S7 TCP-IP Device     | AI1       | AI1          | 4095            | UInt16       |             | 0.ir1          | Good   |
| Modbus Device        | AI2       | AI2          | 4095            | UInt16       |             | 0.ir2,word     | Good   |
| Settings             | AI3       | AI3          | 4095            | UInt16       |             | 0.ir3,uint16   | Good   |
| Control              | AO0       | AO0          | 123999          | UInt32       |             | hr8,dword      | Good   |
| Status               | AO1       | AO1          | 123039          | UInt32       |             | 0.hr10,dword   | Good   |
| Channels             | AO2 + AO3 | AO2 + AO3    | [1040596, 1203] | UInt32-Array |             | 0.hr12,2,dword | Good   |
| Channel 1            |           |              |                 |              |             |                |        |
| Channel 2            |           |              |                 |              |             |                |        |
| Settings             |           |              |                 |              |             |                |        |
| Control              |           |              |                 |              |             |                |        |
| Status               |           |              |                 |              |             |                |        |
| Variables            |           |              |                 |              |             |                |        |

## Logdatei

Alle kanalbezogenen Statusinformationen werden auch in die kanalspezifische Logdatei im [LoggingFolder] protokolliert. Jede Logdatei wird nach dem Namensschema Modbus Device.<ChannelName>.log benannt.

Der Inhalt einer solchen Logdatei kann wie folgt aussehen:

```
...
2018-04-11 11:32:37.0 +2: [Error] Error (Severity=High): Code=[-1],
Text=[The operation has timed-out.], Details=[]
...
```

## Entities



Wie jedes Device Plugin erweitert das Modbus Device Plugin das CoDaBix [Device Modell](#).

## Device

Der Device Typ `ModbusDevice` des Plugins definiert auch den `ModbusDeviceChannel` und erweitert somit die grundlegenden `CodabixDevice` und `CodabixDeviceChannel` Entities. Während das `ModbusDevice` nur eine Konkretisierung des `CodabixDevice` darstellt, erweitert der `ModbusDeviceChannel` den `CodabixDeviceChannel` mit den Modbus Variable Entities.

## Channel

Der Kanal wird von einem Channel Worker behandelt, der eine TCP Socket Verbindung zum lokalen Server herstellt. Zur Fehlerdiagnosezwecken überprüft der Worker automatisch alle 10 Sekunden den Status der TCP Socket Verbindung, um den Channel Statuscode und seine Beschreibung zu aktualisieren und Verbindungsfehler zu erkennen.

Der Worker liest standardmäßig keine Werte. Wenn ein Anwender oder Plugin in CoDaBix® einen synchronen Lesevorgang der Channel Variablen anfordert (z.B. mit der „Read actual value“-Funktion in der CoDaBix® Webkonfiguration), liest der Channel Worker diese vom Modbus Server und schreibt diese in die entsprechenden CoDaBix Nodes.

Ähnlich schreibt der Channel Worker auch die Werte in den Modbus Server, wenn ein Client oder Plugin Werte in die Channel Variablen schreibt.

Damit eine Modbus Variable regelmäßig gelesen wird, können Sie in der Webkonfiguration bei dem Node „History Options“ auf Yes stellen (was eine interne Subscription erstellt), oder Sie können zum Beispiel einen OPC UA Client verwenden, der mit dem OPC UA Serverplugin verbunden ist und damit eine Subscription für die Modbus Variablenodes erstellen. In diesen Fällen liest der Channel Worker in regelmäßigen Intervallen die Variablen vom Modbus und schreibt den neuen Wert nach einer Wertänderung automatisch in die entsprechende CoDaBix® Node.

## Variable

Der Modbus-Datentyp bestimmt, auf welche Modbus-Speicherbereich zugegriffen wird. Dabei werden die Variablenformate Skalar und Array unterstützt.

## Ordner & Dateien

### Ordner

| Name           | Pfad                                            | Zweck / Verwendung                         |
|----------------|-------------------------------------------------|--------------------------------------------|
| AssemblyFolder | <CodabixInstallDir>/plugins/ModbusDevicePlugin/ | Beinhaltet die Plugin Assembly Datei.      |
| ConfigFolder   | <CodabixDataDir>/plugins/ModbusDevicePlugin/    | Beinhaltet die Plugin Konfigurationsdatei. |

| Name          | Pfad                  | Zweck / Verwendung                 |
|---------------|-----------------------|------------------------------------|
| LoggingFolder | <CodabixDataDir>/log/ | Beinhaltet die Plugin Log Dateien. |

## Dateien

| Typ      | Pfad                                            | Zweck / Verwendung         |
|----------|-------------------------------------------------|----------------------------|
| Assembly | [AssemblyFolder]/CoDaBix.ModbusDevicePlugin.dll | Die Plugin Assembly Datei. |
| Logging  | [LoggingFolder]/Modbus Device.<ChannelName>.log | Die Log Datei.             |

## Versionsinformation

### Dieses Dokument

|                |            |
|----------------|------------|
| <b>Datum</b>   | 2023-10-31 |
| <b>Version</b> | 1.1        |

### Plugin

|                |                               |
|----------------|-------------------------------|
| <b>Name</b>    | Modbus Device Plugin          |
| <b>Node</b>    | /System/Devices/Modbus Device |
| <b>Version</b> | 1.0.0                         |

### Assembly

|                |                                |
|----------------|--------------------------------|
| <b>Name</b>    | CoDaBix.ModbusDevicePlugin.dll |
| <b>Datum</b>   | 2018-11-23                     |
| <b>Version</b> | 1.0.0.0                        |

From:

<https://codabix.de/> - **CoDaBix®**

Permanent link:

<https://codabix.de/de/plugins/device/modbusdeviceplugin>

Last update: **2023/11/03 13:10**