Node-Red

straton user guide – Rev. 2

www.copadata.com/embedded

sales.fr@copadata.com

Content

1. Overview

This document describes how to start with the straton Node-Red nodes.
The aim of these nodes is to enable easy interaction between a straton runtime and other Node-Red nodes.

2. Requirement and setup

- For system running the Node-Red framework.

  Any OS that supports the Node-Red framework (MAC-OS, Windows, Linux, Raspberry-PI, etc…)

  .

- For system running the straton runtime

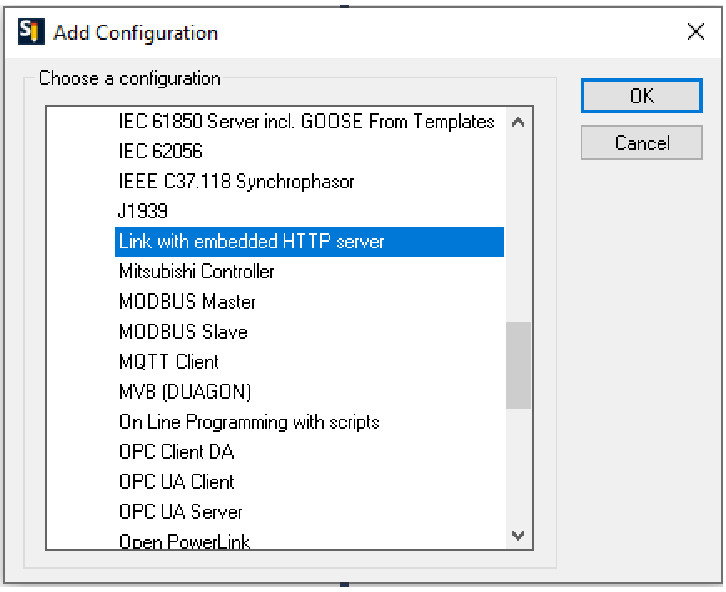
  - The OS must be Linux
  - An HTTP server must be running and supports WebSocket (ex: Apache or Nginx)
  - The straton data-server WebSocket must be running
  - The straton runtime must run the "Link with Embedded HTTP server" driver

NOTE: the straton runtime and the Node-Red framework can run on the same system.

3. Configure and run straton runtime side

3.1. Configure straton application

In order to use the straton data-server in the runtime, you must add the configuration : “Link with Embedded HTTP server”

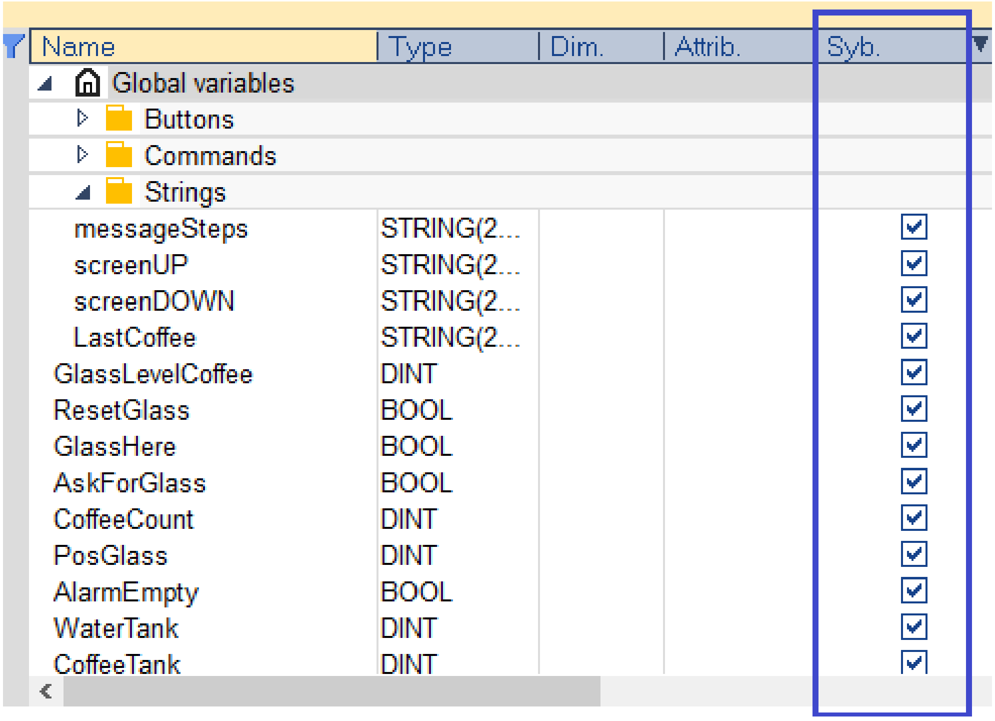


Add Configuration

Choose a configuration

- IEC 61850 Server incl. GOOSE From Templates
- IEC 62056
- IEEE C37.118 Synchrophasor
- J1939
- Link with embedded HTTP server
- Mitsubishi Controller
- MODBUS Master
- MODBUS Slave
- MQTT Client
- MVB (DUAGON)
- On Line Programming with scripts
- OPC Client DA
- OPC UA Client
- OPC UA Server
- Open PowerLink

Be sure to check „Syb“ for each variable you want to use in the data-server.



Name	Type	Dim.	Attrib.	Syb.
Global variables				
Buttons				
Commands				
Strings				
messageSteps	STRING(2...			☑
screenUP	STRING(2...			☑
screenDOWN	STRING(2...			☑
LastCoffee	STRING(2...			☑
GlassLevelCoffee	DINT			☑
ResetGlass	BOOL			☑
GlassHere	BOOL			☑
AskForGlass	BOOL			☑
CoffeeCount	DINT			☑
PosGlass	DINT			☑
AlarmEmpty	BOOL			☑
WaterTank	DINT			☑
CoffeeTank	DINT			☑

3.2. Run the HTTP server and straton WebSocket

- Run the HTTP server (Apache, NGINX, etc…)
- Run the straton WebSocket : ./t5ws
- Run the straton runtime

4. Create a Node-Red flow

4.1. Install straton Node-Red package

On Windows, open a command line.
Go to the Node-Red installation folder, e.g. cd C:\Users\me\.node-red

Type the command: npm install node-red-contrib-straton

4.2. Run Node-Red framework and editor

Run the Node-Red framework:

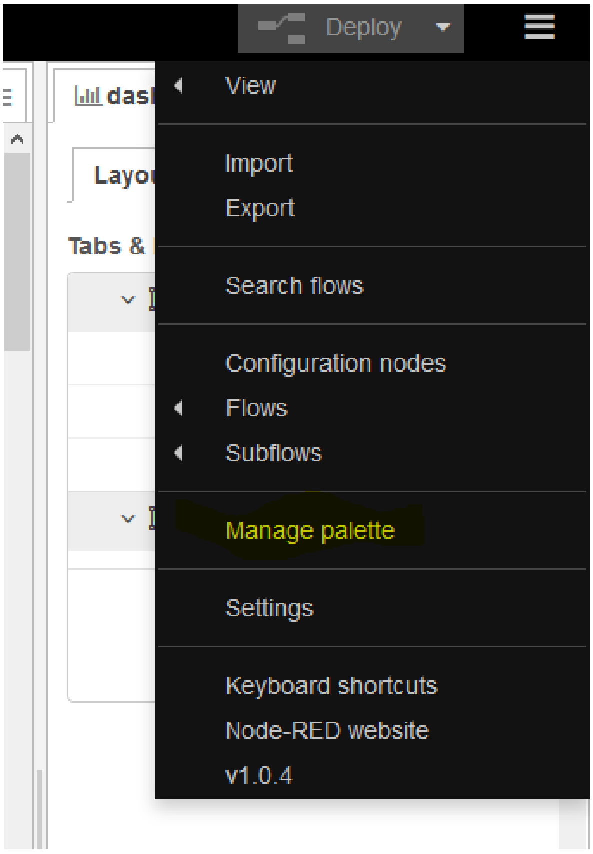
On Windows, open a command line and type: node-red

Open the editor:

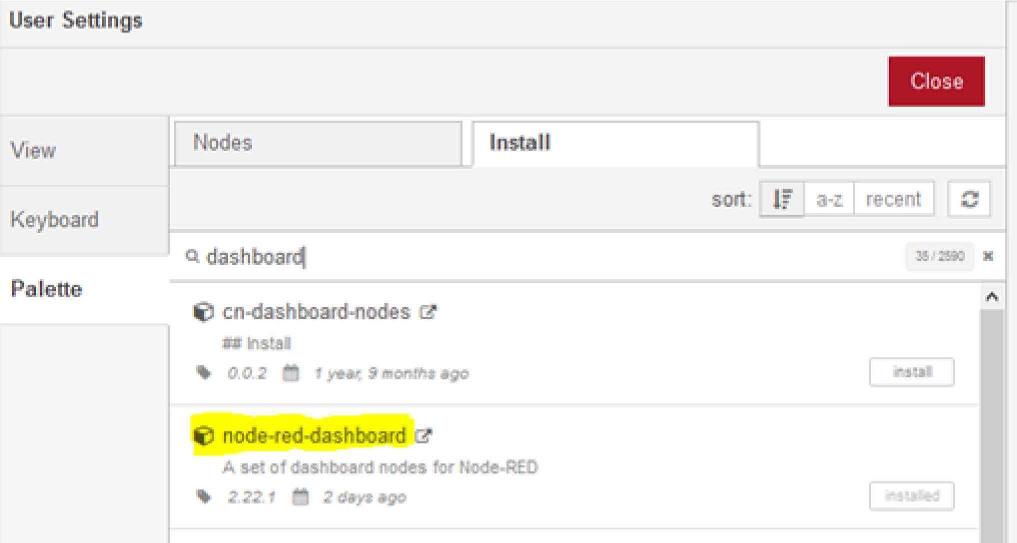
Open a Web browser and go to : http://localhost:1880

Notes:

- By default, the port is 1880. Depending on your Node-Red installation, the port can have another value
- You can access the editor from another computer by entering the IP address oft he PC running the Node-Red framework e.g. , http://192.168.1.1:1880
- By default, the graphical UI nodes like gauge, button, etc… are not installed.
  To install them, go to „Manage Palette"

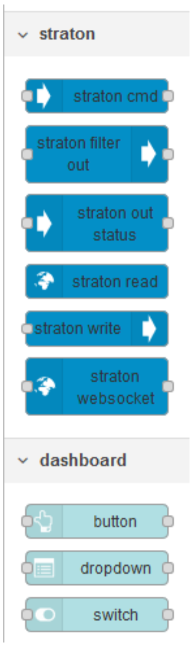


And then select „Install“ tab, enter „dashboard“ in the search area and install the package „node-red-dashboard“



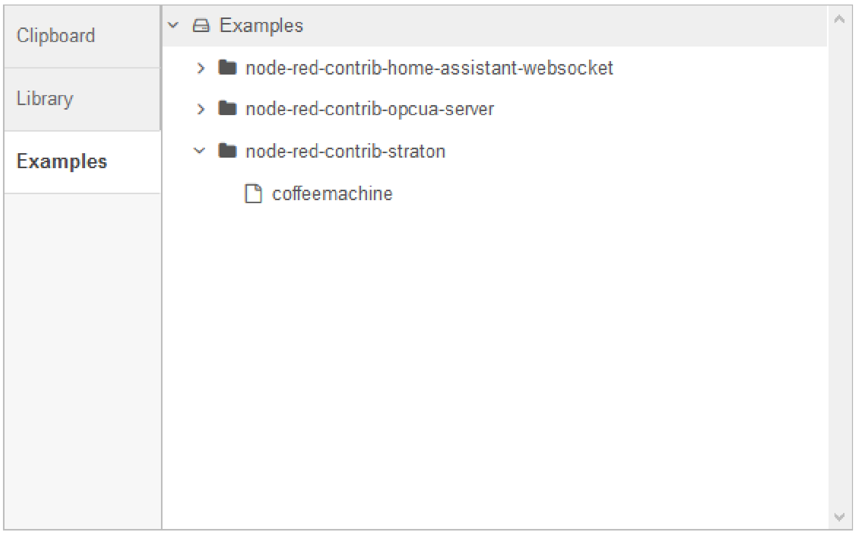
4.3. Create your Node-Red flow

You can use the different straton nodes, located in the category „straton“ in the palette



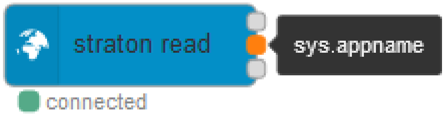
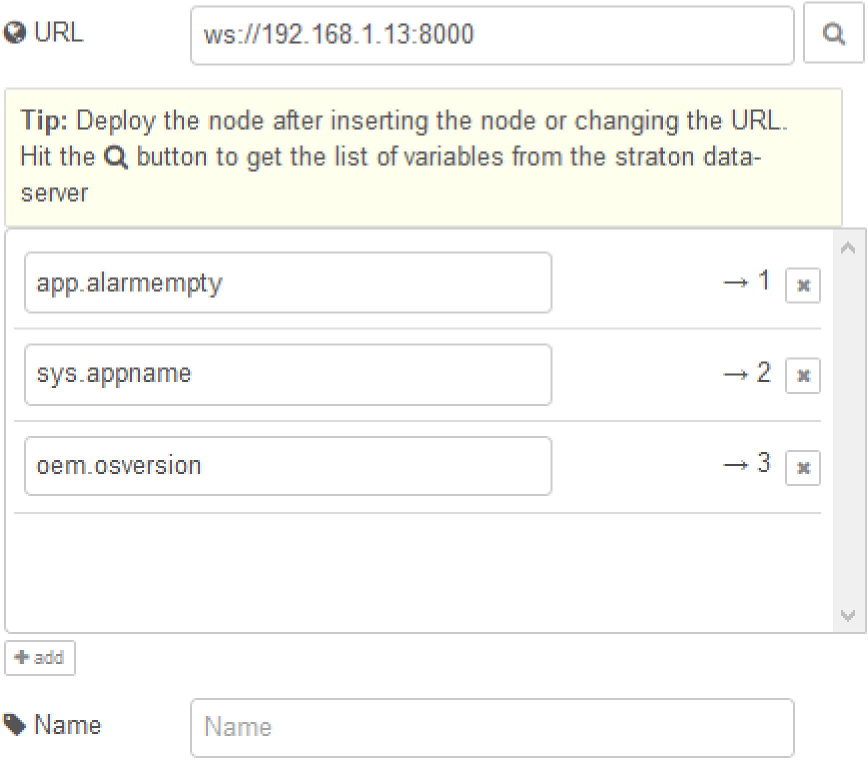
All the straton nodes are described in the chapter 5 of this tutorial.

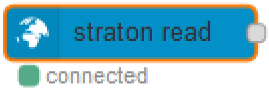
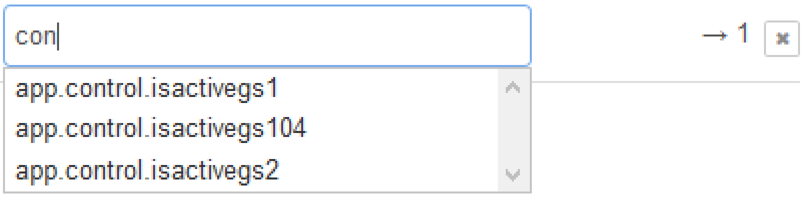
NOTE : Some examples are provided as flows. They are accessible through the import command of the dashboard


Clipboard
Library
Examples
Examples
node-red-contrib-home-assistant-websocket
node-red-contrib-opcua-server
node-red-contrib-straton
coffeemachine

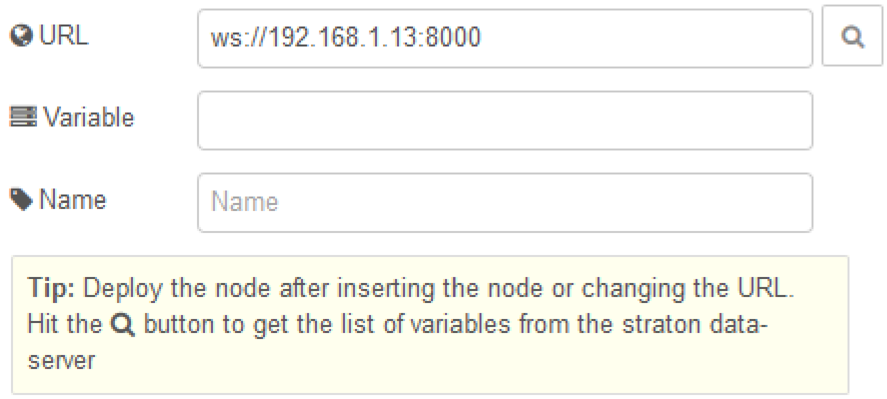
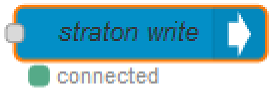
5. Straton Node-Red nodes

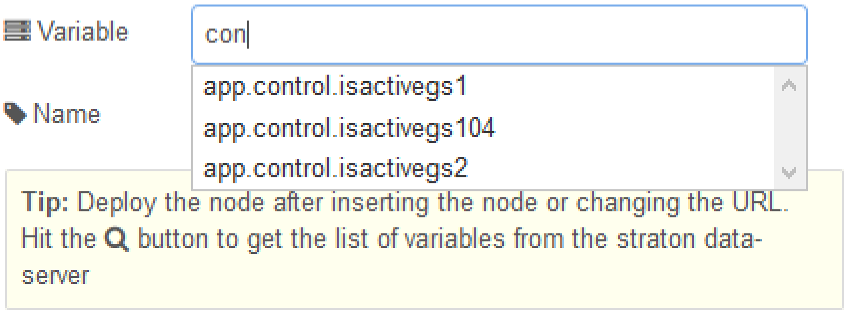
5.1. Straton read

Description	Connect to a WebSocket and subscribe to one or several straton variables. There will be an output for each subscribed variable. Each time the value of a variable is changed, a message with value will be sent on the corresponding output
Input	None
Output	0 to n (0 to n straton variables to read) When you hover an output with the mouse, the corresponding variable name appears in a tooltip 
Node's properties	 URL ws://192.168.1.13:8000 **Tip:** Deploy the node after inserting the node or changing the URL. Hit the 🔍 button to get the list of variables from the straton data-server app.alarmempty → 1 ✖ sys.appname → 2 ✖ oem.osversion → 3 ✖ Name Name

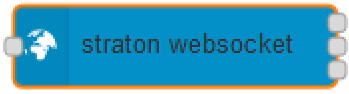
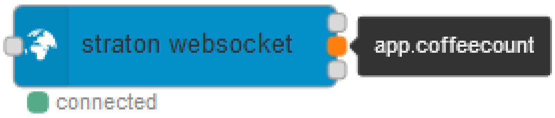
	URL: url of the websocket to connect to List of variables: variables to subscribe to. You can add variable by clicking on + add button. You can remove a variable by clicking on ✖ button. Each straton variable name has the following format: app.xxxx for straton variables sys.xxxx for runtime variables (like straton application name) oem.xxxx for oem variables (like Linux os version) See chapter 7.2 for list of system variables. All the variable names are case insensitive Name: Name of the node to be displayed in the NODE-Red Dashboard (optional)
Notes	- If the node is deployed and an URL has been given, it tries to connect to the WebSocket. A status icon and text will be displayed just below  - If the node is deployed and an URL has been given, you can click on the 🔍 button to get the list of variables available on the straton runtime. An auto-completion will then be available as you type the variable name  con app.control.isactivegs1 app.control.isactivegs104 app.control.isactivegs2 - If the variable name is left blank, the corresponding output won't be created

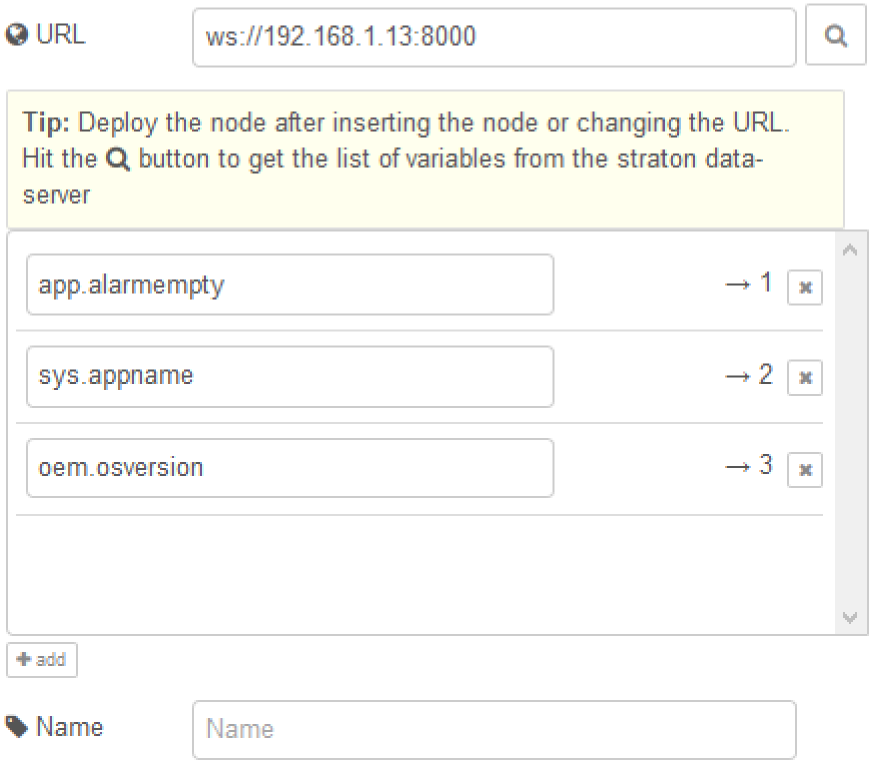
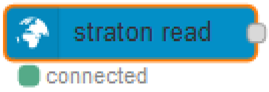
5.2. Straton write

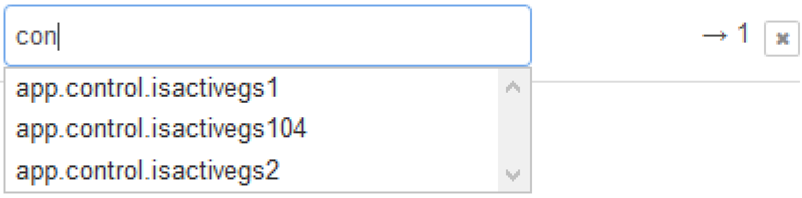
Description	Connect to a WebSocket and write value of a straton variable. Each time a message is coming in, the configured variable will be written
Input	1 The payload of the incoming message must contain the value. The value can be a string or a number.
Output	0
Node's properties	 URL ws://192.168.1.13:8000 Variable Name Name **Tip:** Deploy the node after inserting the node or changing the URL. Hit the 🔍 button to get the list of variables from the straton data-server URL: url of the websocket to connect to Variable: variables to write value to. Name: Name of the node to be displayed in the NODE-Red Dashboard (optional)
Notes	- If the node is deployed and an URL has been given, it tries to connect to the WebSocket. A status icon and text will be displayed just below  - If the node is deployed and an URL has been given, you can click on the 🔍 button to get the list of variables available on the straton runtime. An auto-completion will then be available as you type the variable name

	 Variable: con Name app.control.isactivegs1 app.control.isactivegs104 app.control.isactivegs2 **Tip:** Deploy the node after inserting the node or changing the URL. Hit the 🔍 button to get the list of variables from the straton data-server - If the variable name is left blank, the node cannot be deployed

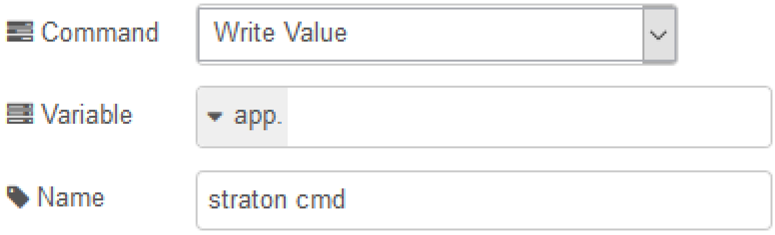
5.3. Straton WebSocket

Description	Connect to a WebSocket and subscribe to one or several straton variables. Additionally, it can send other commands when a message is coming in. There will be an output for each subscribed variable. Each time the value of a variable is changed, a message with value will be sent on the corresponding output. An additional output will send message with the result of the command sent 
Input	1 The payload of the message must be a valid command to be sent to the straton data-server. The command can be a string or a JSON object. It will be converted in string when sent to the data-server. The format of the different commands will be given in annex
Output	0 to n+1 (0 to n straton variables to read + 1 output for other result) The additional output is named "standard" and is always the last one. When you hover an output with the mouse, the corresponding variable name appears in a tooltip 

Node's properties	 URL ws://192.168.1.13:8000 **Tip:** Deploy the node after inserting the node or changing the URL. Hit the 🔍 button to get the list of variables from the straton data-server app.alarmempty → 1 ✖ sys.appname → 2 ✖ oem.osversion → 3 ✖ + add Name: Name URL: url of the websocket to connect to List of variables: variables to subscribe to. You can add variable by clicking on + add button. You can remove a variable by clicking on ✖ button. Each straton variable name has the following format: app.xxxx for straton variables sys.xxxx for runtime variables (like straton application name) oem.xxxx for oem variables (like Linux os version) See chapter 7.2 for list of system variables. All the variable names are case insensitive Name: Name of the node to be displayed in the NODE-Red Dashboard (optional)
Notes	- If the node is deployed and an URL has been given, it tries to connect to the WebSocket. A status icon and text will be displayed just below  🟩 connected

	- If the node is deployed and an URL has been given, you can click on the 🔍 button to get the list of variables available on the straton runtime. An auto-completion will then be available as you type the variable name  con app.control.isactivegs1 app.control.isactivegs104 app.control.isactivegs2 - If the variable name is left blank, the corresponding output won't be created

5.4. Straton command

Description	Send a data-server command to its output when receive a message. This node should be used in conjunction with Straton Websocket.
Input	1 The message is used as a trigger to send output. Additionally, if the command is a Write, the payload should contain the value in string or number format. In other case, the payload is not used.
Output	1 The payload of the message contains the command in a JSON object
Node's properties	 Command Write Value Variable app. Name straton cmd **Command:** command to be generated **Variable:** variable to read or write. Each straton variable name has the following format: app.xxxx for straton variables sys.xxxx for runtime variables (like straton application name) oem.xxxx for oem variables (like Linux os version) See chapter 7.2 for list of system variables. All the variable names are case insensitive **Name:** Name of the node to be displayed in the NODE-Red Dashboard (optional)
Notes	- The field 'Variable' appears only if the command is Write or Read. - As the node does not connect to the WebSocket, it cannot retrieve variable list. Auto- completion is not available for this node. - If the variable name is left blank, for Read or Write command, the node cannot be deployed.

6. Straton Data-Server commands

In this chapter, are described the different commands that can be sent to the Data-Server.

Each command is described in JSON format.

6.1. Get list of symbols/variables

Description	returns the list of symbols of the application
Call (payload)	{"cmd":"list2"}

Response	{
	"cmd":"list2",
	"payload": ["var1","var2","var2"],
	"statusCode":200,
	}

6.2. Get runtime status

Description	returns the list of symbols of the application
Call (payload)	{"cmd":" status2"}

Response	{
	"cmd":"status2",
	"payload":
	{
	"AppName":"CoffeeMachine",
	"AppVer":"121",
	"CycleTime":"0",
	"CycleMax":"1",
	"CycleOvf":"0",
	"CycleCount":"18813148",
	"Flags":"37889"
	},
	"statusCode":200,
	}

6.3. Read

Description		Read a variable

Call (payload)		{
		“cmd”:”read2”,
		”data”:[{“name”:”app.watertank”}]
		}

Response		{
		"cmd":"read2",
		"payload":
		[
		{
		"name":"app.watertank",
		"value":"1560",
		"error":200,
		"extError":0
		}
		],
		"statusCode":200
		}

6.4. Write

Description	write variable

Call (payload)	{
	"cmd":"write2",
	"data":[{"name":"app.watertank","value":"23"}]
	}

Response	{
	"cmd":"write2",
	"payload":
	[
	{
	"name":"app.watertank",
	"error":200,
	"extError":0
	}
	],
	"statusCode":200
	}

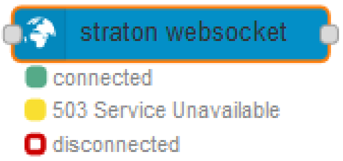
7. Frequently Asked Questions

7.1. Node status

For all the nodes that connect to a WebSocket, they should be able to re-connect if the connection is lost.

If the runtime is stopped and re-started, the nodes should be able to re-connect also.

In both cases, the status of the node should reflect the status of the connection in the Dashboard



straton websocket

🟩 connected

🟨 503 Service Unavailable

🟥 disconnected

7.2. System variables

Name	Description
sys.AppName	Straton application name
sys.AppVer	Straton application version
sys.CycleTime	Current cycle time
sys.CycleMax	Maximum cycle time
sys.CycleOvf	Number of cycle overflows
sys.CycleCount	Number of cycles
sys.Flags	Straton application flags

7.3. Error codes

All the error codes coming from the straton Data-Server are HTML error codes and described below:

Value	Description
400	ERROR_BADREQUEST
500	ERROR_INTERNALSERVERERROR
501	ERROR_NOTIMPLEMENTED
502	ERROR_BADGATEWAY
503	ERROR_SERVICEUNAVAILABLE

504	ERROR_GATEWAYTIMEOUT
ERROR_NOERROR	200
ERROR_UNKNOWNERROR	520