# Nodes

## HDL Bus

The HDL bus node provides direct connection to the HDL bus.

### Settings

|  |  |  |
| --- | --- | --- |
| ***Setting*** | ***Description*** | ***Possible values*** |
| Name | The name of the node that will appear in the flow | String |
| Output mode | The amount of information that will be outputted. | * Only output answer back (Only output when a message is being replied to from this PC) * Output all messages sent to this device (Outputs everything that is sent to the local address of this PC) * Output all messages (Outputs everything on the bus) |
| Network | The HDL network device that is providing connection to the bus via a network connection | Hdl-network node |

### How to use

In general, the node expects the same information as it outputs. The node expects the general msg object described in the documentation under general layout of the msgs.

# General layout of the msgs

The nodes expect a general layout of the msg object. This can differ depending on the command sent and if the required command is supported.

The following is the basic layout of the parameters that can be input/output by the nodes:

* ***msg***

- ***payload***

- opCode (The operate code defined in the HDL documentation)

- sender (The device that sent the message)  
 - deviceType

- subnetId

- deviceId

- wasSentToThisDevice

- operate  
- mode  
- direction  
- data  
- contents (The raw buffer of the data)

- ***node*** (This is the information of the node that sent out the message)

- name

- type

- outputMode

If the command is supported the operate, mode, direction, and data parameters will not be null when receiving these will contain processed data that will be explained in detail in their respective documentation.

When sending there are two ways to send the information, first send it using the supported commands, or send a raw packet.

1. Supported method (Scene control as an example)

var msg = {

“payload”: {

“operate”: “sceneControl”,

“mode”: “set”,  
 “direction”: “request”,

“data”: {

“areaNumber”: 1,

“sceneNumber”: 1

}

}

}

return msg;

The above message will set the scene of the device is was sent to scene 1 in area 1.

1. Sending a raw packet (Advanced)

var msg = {

“payload”: {

“opCode”: 0x0002,

“contents”: Buffer.from([1, 1])

}

}

return msg;

The above message will do the same, set the scene of the device is was sent to scene 1 in area 1.

When a message is received on the bus the node will output the entire message stated above. However if the command is not supported the operate, mode, direction, and data parameters will be null.