A2Z-Link

User Manual

Prepared by: Administrator
Date: 3/27/2015

A2Z-Link

Table of Contents

A2Z-Link

Online **Help**

Welcome to the A2Z-Link online help system. Browse through the help pages by clicking on the icons below or selecting pages in the table of contents to the left. To quickly find specific product information, enter search criteria in the search box above and click the search button.

Getting Started

ASCII Control Interface

JSON Control Interface

Ask Us

If you're unable to find what you're looking for in this help system, try these alternative resources:

- Our Website
- Knowledgebase
- FAQ

or contact our team:
Email: sales@homeseer.com
Phone: 603-471-2816

Most popular pages

Getting Started

Articles in this section


System Overview


Installation


Managing Z-Wave

See also

Welcome
ASCII Control Interface
JSON Control Interface

System Overview

What is A2Z-Link?

A2Z-Link is a hardware module that allows systems to communicate with Z-Wave devices without any knowledge of Z-Wave technology. A Z-Wave system can be very complex as it comprises many command classes that are used to control specific devices. The A2Z-Link uses simple "ASCII" or text commands to control Z-Wave devices and gather their status. The device supports 2-way communication, so status may also be maintained. The unit supports either simple text commands, or more robust JSON for communications. Either or both protocols may be used.

A2Z-Link Features:

- Can Function as either a primary or secondary controller
- Supports Replication (send and receive)
- Supports multiple Z-Wave networks using multiple units (allows for installation in any size premises, including support for outbuildings)
- Each unit supports up to 232 Z-Wave devices
- Includes diagnostics for connectivity and network healing (healing is the ability to tell a device that it needs to update its neighbor table, this is required for proper command routing in the Z-Wave mesh network)
- Supports associations for instant status
- Supports polling for device status for nodes that do not support associations
- Includes Add and Remove functions for Z-Wave nodes
- Provides detailed information for all Z-Wave nodes, including node capabilities, firmware version, etc.
- Supports firmware update class for supported devices
- Includes device management web interface for full device control, testing, configuration, scene management, etc.
- Includes configurable IP port number for control
- Supports simple ASCII command or JSON control protocol
- Provides full 2-Wave control of devices with instant status report
- Includes Z-Wave scene support

Why HomeSeer?

HomeSeer was the first control system company to support Z-Wave back in 2003. Today, our software is the culmination of more than a decade's worth of Z-Wave engineering experience. We work very hard to support new Z-Wave command classes and products, as they are released; this is a primary focus for us. Our Z-Wave inclusion feature employs a robust 'interrogation' process that's designed to uncover and provide support for all aspects of a product's feature set, including special parameter settings. This allows installers to adjust devices in many ways. Diagnostics features are also included to provide connectivity testing and network healing functions.

Other technologies

A2Z-Link can also be used to provide integration with these technologies:

- Insteon
- UPB (Universal Powerline Bus)
- X10
- Philips Hue
- Any other system supported by the HomeSeer control software (contact HomeSeer for details)

See also
Installation
Managing Z-Wave

Installation

Installing A2Z-Link

The unit connects to the control system via a network connection and all communications are done over IP. Power is applied via the included micro USB power adapter. The unit will boot up automatically when power is applied; there is no power switch.

1. Apply power and wait approximately 2 minutes. The LED on the unit will flash as the unit starts up.
2. The unit uses DHCP to obtain an IP address, to find the unit on the network, bring up a web browser on any computer that is connected to the same network. In the URL enter: http://find.homeseer.com
3. Click the Search button. The unit should then be listed; click on the link to connect to the unit.
4. Please refer to the Z-Wave section of this guide for information on managing Z-Wave devices.
5. For remote connections, myHomeSeer Remote can also be used. Create an account at http://myhs.homeseer.com. All you need is the license ID and password printed on the bottom of the unit. Once the account is created, you can log into the unit remotely at http://myhs.homeseer.com. If myHomSeer Remote is not to be used, it can be disabled from the Network tab in Setup.

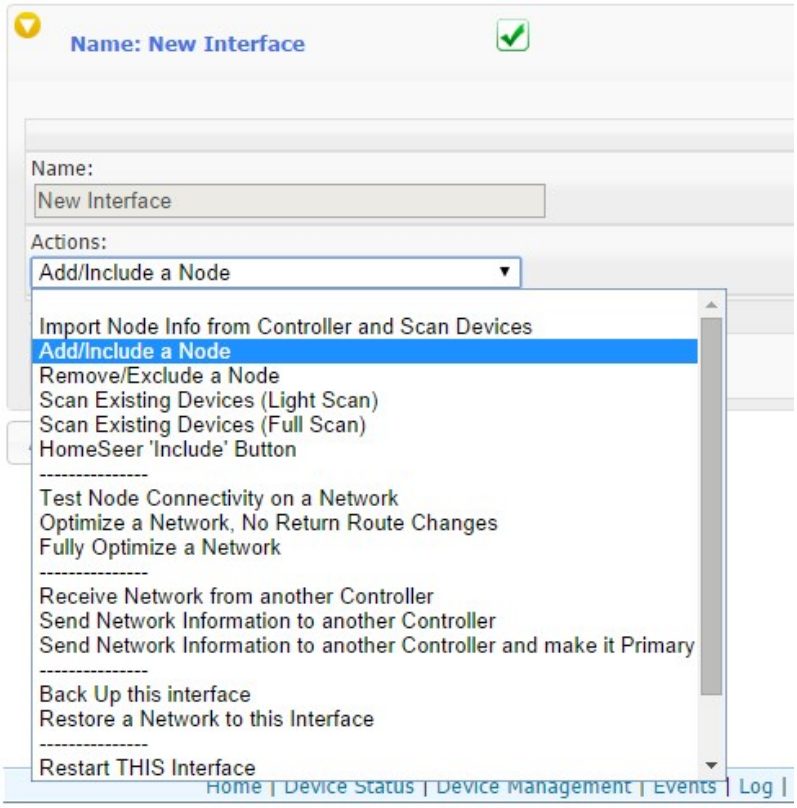
See also
System Overview
Managing Z-Wave

Managing Z-Wave

New Z-Wave Installation Using A2Z-Link

If a Z-Wave network has not yet been established at the installation site, this may be done with A2Z-Link. To add Z-Wave devices, navigate to the A2Z-Link web interface, select the "PLUG-INS" menu, then Z-Wave, then Controller Management. Expand Z-Wave Interfaces and select "Add/Include a Node" from the Actions drop list. A log window will appear with a "Start" button just above it. Progress will be displayed on the screen as well as any instructions. Click the Start button and then activate the inclusion process on the device you are adding. Typically, this is done by pressing a button or switch paddle or by following a sequence of steps (as with many door locks). Consult with the manufacturer's documentation for specific steps, if you encounter problems with this*. A message will display after the device adds successfully. Click the "Close" button to close the log window when finished.

*Note: If the device was previously added to another network, it will not add to this network until it has been reset. To reset the device follow the procedure above but select the "Remove/Exclude a Node" function instead of "Add/Include a Node". Once the device has been successfully removed (reset), you should be able to add it successfully.



Controller Management "Actions" Menu

Adding A2Z-Link to Existing Z-Wave Installations

If the site already has an existing Z-Wave network a decision will need to be made.

1. If the devices were added using a simple handheld controller, or if the devices were added using a controller that will no longer be used, you should consider establishing a new network with the A2Z-Link. This will allow the A2Z-Link to function as the primary Z-Wave controller and you'll have many more Z-Wave management options available going forward (including replication and optimization functions). To do this, you'll need to reset (remove) all nodes from the existing network and then establish a new network using A2Z-Link. Follow the procedure outlined above.
2. If the existing controller is to remain and (a) you wish to keep it as a primary controller or (b) it cannot function as a secondary controller, then A2Z-Link can be added to your existing network as a secondary controller. To make the A2Z-Link a secondary, the primary controller needs to be put in Replication Send mode, then select "Receive Network from another Controller" from the Controller Management "Actions" menu in the A2Z-Link web interface. Progress information will be displayed. Note that A2Z-Link cannot perform replication or optimization (healing) functions when operating as a secondary controller.

See also

System Overview
Installation

ASCII Control Interface

Articles in this section

Control

Status

See also

Welcome
Getting Started
JSON Control Interface

Control

Connection

Commands are sent over a TCP connection. The connection may remain open, or it may be opened and closed as needed. The default TCP port is port # 11000. To test, start a Telnet session on a computer connected to the same network as the A2Z-Link and connect to port 11000. This command is typically:

telnet ip-address-of-a2zlink 11000

Then enter:

vr[ENTER]

A version # should be returned such as:

3.0.0.127

Connections need to be authorized. In Setup on the Network tab users may be set up. A user with rights Admin or Normal will have access. Guests will not have access. The connection IP is authorized with the following command:

au,user,pass

If the user is authorized, the command returns "ok". After authorization, commands may be given from the source IP address. To de-authorize a connect use the command:

lo

This will remove the source IP address from the this list of authorized IP addresses.

Controlling Devices

The A2Z-Link abstracts Z-Wave devices so there is no need to understand the intricacy of the Z-Wave protocol. When devices are added, the A2Z-Link creates virtual devices that represent the functionality of the target device. For example. if a simple Z-Wave lamp module is added, it will be viewed on the Device Management web page like:

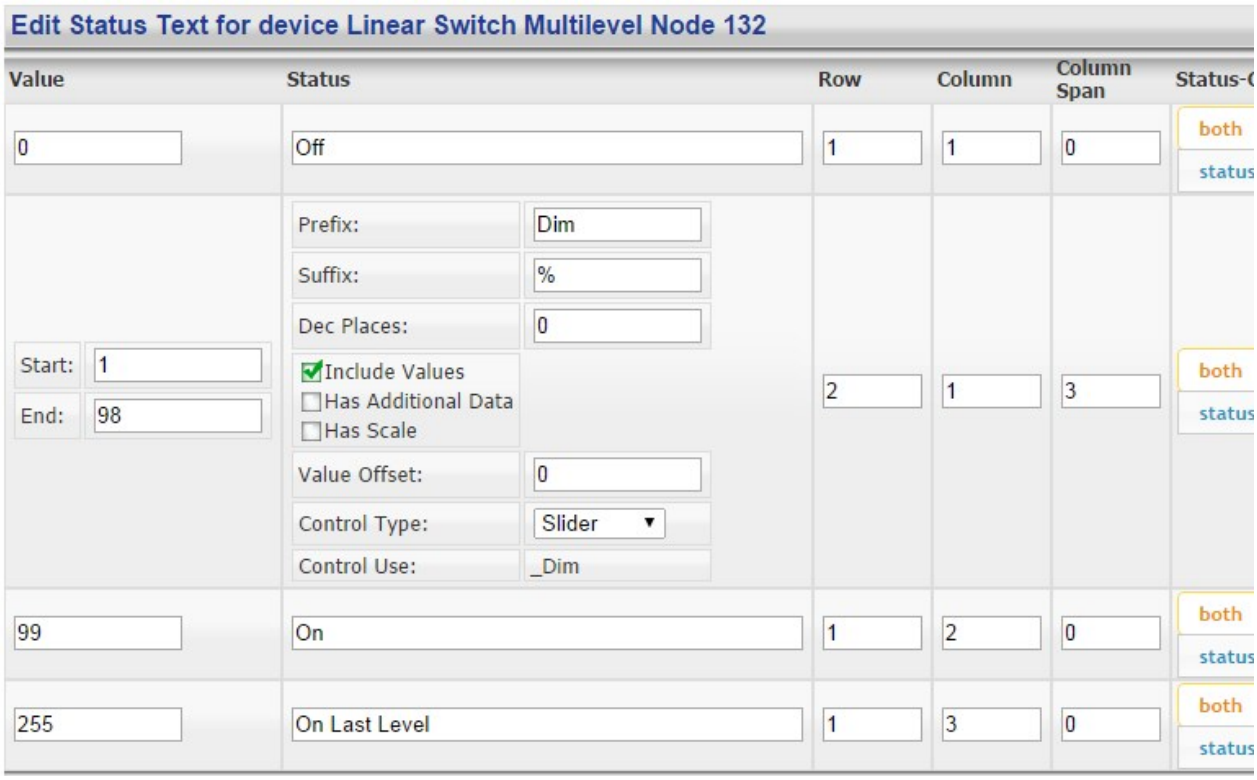
Ref	Status	Room	Floor	Name	Last Change
3738	Off	Z-Wave	Node 132	Linear Switch Multilevel Node 132	

This virtual device represents the lighting control functionality of the device and shows it can be turned On,Off, and Dimmed. These devices are controlled with their reference #. A reference # is a unique ID assigned to the device and cannot be changed. Other properties of the device may be changed as needed, such as the location and name. The reference # is listed in the second column as can be seen above.

Devices are controllable by setting their value or label. Each device holds a collection of name/value pairs that represent the available control points. Control points can be a single value, such as 0 for Off, or a value range such as 1-98 for a dim level. In the web interface on the device management page you can click on the name of a device to bring up its properties. Click on the "Status/Graphics" tab to see the available pairs. From here you can determine how to control the device. You can also issue a command to get the pairs if your control system can deal with that.

Commands are available to retrieve the control information if your control system can support this.

Here is an example of the status pairs for a dimmable light. It can be seen that the possible control values are 0 for Off, 99 for On, 255 for on last level, and 1-98 for the dim level.



Commands are issued as simple text strings terminated with a carriage return and linefeed.

Commands are not case sensitive.

Parameters in [] are optional.

The following commands are available for device control:

Command	Format	Description	Example
Control a device by value	cv,ref,value	Controls a device by settings its value. All devices have a collection of name/value pairs that represent controllable parts of the device, such as On=99 and Off=0. The value is used for control.	Turn on a light where the reference # is 2356 and the ON value is 99: cv,2356,99

Control a device by label	cl,ref,label	Controls a device using the control label. All devices have a collection of name/value pairs that represent controllable parts of the device, such as On=99, and Off=0. The actual label is used for control.	Turn on a light where the reference # is 2356 and the ON label is On: cl,2356,On

Retrieving control information

For advanced systems that can dynamically create a user interface, the control information can be obtained. A request may contain a single device reference #, of, if omitted, control information for all devices is returned.

Command	Format	Description	Example
Get control information	gc,[ref]	Returns control information for a given device, all all devices if the reference # is omitted. The return format is: ref,label=value,label=value,... If the device supports a control point that accepts a range of values, such as dimming light, the control pair will be returned as: PREFIX (value)SUFFIX=FROM->TO The PREFIX is a label that appears before the value and the SUFFIX is a label that appears after the value. The PREFIX and SUFFIX are listed on the "Status/Graphics" tab in the device properties and can be changed if needed. **Note**: If a label contains a "," it will be escaped and returned as "\,".	For example, a dimming device with ref 3755 may have its PREFIX set to "Dim" and its SUFFIX set to "%", and the valid values are 1-98, the return would be: 3755,On=99,Off=0,Dim (value)%=1->98

See also
Status

Status

Please read the Control section first for information on how to connect to the A2Z-Link and how to format commands.

For more advanced control systems and control systems drivers, you may want to dynamically retrieve the available devices in the system and get their current status. The following commands can be used to discover all devices, get their current status, and assigned name and location. The A2Z-Link allows for 2 location parameters to be set on each device, as well as a device name. This information can be maintained in the A2Z-Link itself, or left off and maintained by the controller.

Command	Format	Description	Example

Get device status.	gs,[ref]	Returns a comma list of devices. If the ref parameter is omitted, all devices are returned seperated with a "|", else just information for the requested ref number. Format of return is: ref,parent_ref,status,name,location2,location1 Where: ref = unique reference # of this device, will match request if a specific ref is entered in the request. parent_ref = unique reference # of parent of this device. Some devices, such as thermostats, comprise many devices. One device will be assigned as the parent and is used to note which physical device this device belongs to. If the device does not have a parent, which means it is either the root device, or it is a device that has no children, the value be 0. name = the name of the device location2 = location 2 assigned to this device location1 = location 1 assigned to this device **Note**: If the name, location1, or location2 contains a "," it will be escaped and returned as "\,".	Request the status of a light switch with ref 3755, the return might be: 3755,0,off,Lights,First Floor,Kitchen

Notification of updates

If a device changes status, a notification is sent over the open port. The format is:

DC,REF,NEWVAL,OLDVAL

For example, the device with reference # 576 changes from 0 to 1:

DC,576,1,0

See also

Control

JSON Control Interface

Articles in this section

Controlling with JSON

See also

Welcome
Getting Started
ASCII Control Interface

Controlling with JSON

The A2Z-Link can be controlled using JSON formatted data. This can be used to get status from devices as well as control devices.

The requests and posts are sent to the unit on port 80.

Some notes on this API:

* Parameters are not case sensitive
* most parameters accept "all" as an option to include all items, or if the parameter is not included "all" is assumed

The following commands are available:

GET requests

URL	Description
	Returns the status of a device in the system or all the devices in following format. The following parameters are supported in or filter the response: location1=loc1 (only return the devices that are in the specific location1, omit or set to "all" for all devices at this location) location2=loc2 (only return the devices that are in the specific location2, omit or set to "all" for all devices at this location) ref=## (only return the device that matches the specific refere this may be a list of reference #'s like 3467,2342,869, omit or "all" to return all devices) All GET requests are terminated with a CRLF. If no JSON data is expected from the request, the return may b "ok", or "error". This response is for 2 devices, the first is a thermostat tempera the second is a light switch: {

/JSON?
request=getstatus&ref=##&location1=LOC1&location2=LOC2


```
"Name":"HomeSeer Devices",
"Version":"1.0",
"Devices":[
   {
      "ref":3398,
      "name":"Temperature",
      "location":"Z-Wave",
      "location2":"Node 122",
      "value":82,
      "status":"82 F",
      "device_type_string":"Z-Wave Temperatu
      "last_change":"\/Date(1410193983884)\/"
      "relationship":4,
      "hide_from_view":false,
      "associated_devices":[
         3397
      ],
      "device_type":{
         "Device_API":16,
         "Device_API_Description":"Thermosta
         "Device_Type":2,
         "Device_Type_Description":"Thermostat
         Temperature",
         "Device_SubType":1,
         "Device_SubType_Description":"Tempe
      },
      "device_image":""
   },
   {
      "ref":3570,
      "name":"Switch Binary",
      "location":"Z-Wave",
      "location2":"Node 124",
      "value":255,
      "status":"On",
      "device_type_string":"Z-Wave Switch Bi
      "last_change":"\/Date(1410196540597)\/"
      "relationship":4,
      "hide_from_view":false,
      "associated_devices":[
         3566
      ],
      "device_type":{
         "Device_API":4,
         "Device_API_Description":"Plug-In A
         "Device_Type":0,
         "Device_Type_Description":"Plug-In
         0",
         "Device_SubType":37,
         "Device_SubType_Description":""
      },
      "device_image":""
   }
]
}
```


Where:

ref = unique device reference number, used for any subsequent requests such as device control, leave blank or set to "ALL" to g status for all devices
name = the name of the device
location = the location of the device such as "kitchen"
location2 = the second location of the device such as "first flo

value = the current value of the device, a double
status = the current string that represents the status of the de
such as "on" or "off"
last_change = the date/time the device last changed status
relationship = 2:root device (other devices may be part of this
physical device),3:standalone=this is the only device that repre
this physical device,4:child=this device is part of a group of de
that represent this physical device
hide_from_view = true/false if true, this device has been set
be visible in any user interface
device_type_string = The string the describes this device, se
plugin authors on their devices.
associated_devices = a list of device reference #'s that are
associated with this device. If the device is a ROOT device, then
is child devices, if the device is a child device, then the list will c
one device that is the root device.
device_type = Detailed information about the capabilities of the
device, used mainly to determine if the device adheres to other
such as thermostat, energy, etc. See the Devices section in the S
section in the HomeSeer HS3 user documentation for informatio
the properties in this section.

Returns control information for a device in the system, or all de
Devices contain "control pairs", one pair for each possible contr
For example, a light that can be turned on and off would contai
control pairs, one for ON and one for OFF. The control pair des
how to control the device. The most important information is th
and the "ControlValue", as those will be needed when the devic
be controlled. The information from this call can be used to bui
user interface that will control the device.

Parameters:

ref=### (where ### is the device reference #, or "all" to return
information for all devices)


```
{
   "ControlPairs":[
      {
         "Do_Update":true,
         "SingleRangeEntry":true,
         "ControlButtonType":0,
         "ControlButtonCustom":"",
         "CCIndex":0,
         "Range":null,
         "Ref":2256,
         "Label":"On Last Level",
         "ControlType":5,
         "ControlLocation":{
            "Row":1,
            "Column":3,
            "ColumnSpan":0
         },
         "ControlLoc_Row":1,
         "ControlLoc_Column":3,
         "ControlLoc_ColumnSpan":0,
         "ControlUse":4,
         "ControlValue":255,
         "ControlString":"",
         "ControlStringList":null,
         "ControlStringSelected":null,
         "ControlFlag":false
      },
      {
```

/JSON?request=getcontrol&ref=##


```
      "Do_Update":true,
      "SingleRangeEntry":true,
      "ControlButtonType":0,
      "ControlButtonCustom":"",
      "CCIndex":1,
      "Range":null,
      "Ref":2256,
      "Label":"On",
      "ControlType":5,
      "ControlLocation":{
         "Row":1,
         "Column":2,
         "ColumnSpan":0
      },
      "ControlLoc_Row":1,
      "ControlLoc_Column":2,
      "ControlLoc_ColumnSpan":0,
      "ControlUse":1,
      "ControlValue":99,
      "ControlString":"",
      "ControlStringList":null,
      "ControlStringSelected":null,
      "ControlFlag":false
   },
   {
      "Do_Update":true,
      "SingleRangeEntry":true,
      "ControlButtonType":0,
      "ControlButtonCustom":"",
      "CCIndex":2,
      "Range":null,
      "Ref":2256,
      "Label":"Off",
      "ControlType":5,
      "ControlLocation":{
         "Row":1,
         "Column":1,
         "ColumnSpan":0
      },
      "ControlLoc_Row":1,
      "ControlLoc_Column":1,
      "ControlLoc_ColumnSpan":0,
      "ControlUse":2,
      "ControlValue":0,
      "ControlString":"",
      "ControlStringList":null,
      "ControlStringSelected":null,
      "ControlFlag":false
   },
   {
      "Do_Update":true,
      "SingleRangeEntry":true,
      "ControlButtonType":0,
      "ControlButtonCustom":"",
      "CCIndex":3,
      "Range":{
         "RangeStart":1,
         "RangeEnd":98,
         "RangeStatusDecimals":0,
         "RangeStatusValueOffset":0,
         "RangeStatusDivisor":0,
         "ScaleReplace":"",
         "HasScale":false,
```

```
            "RangeStatusPrefix":"Dim ",
            "RangeStatusSuffix":"%"
         },
         "Ref":2256,
         "Label":"Dim (value)%",
         "ControlType":7,
         "ControlLocation":{
            "Row":2,
            "Column":1,
            "ColumnSpan":3
         },
         "ControlLoc_Row":2,
         "ControlLoc_Column":1,
         "ControlLoc_ColumnSpan":3,
         "ControlUse":3,
         "ControlValue":1,
         "ControlString":"",
         "ControlStringList":null,
         "ControlStringSelected":null,
         "ControlFlag":false
      }
   ],
   "ref":2256,
   "name":"light",
   "location":"Office",
   "location2":"First Floor"
}
```


Where:

Label = The label to display on this control, if its a button, it w
the button label like "On" or "Off"
ControlType = Specifies the control type for this control, poss
values are:

```
 Not_Specified = 1
       Values = 2                    'This is the default to use if o
the others is not specified.
       Single_Text_from_List = 3
       List_Text_from_List = 4
       Button = 5
       ValuesRange = 6               'Rendered as a drop-list by
       ValuesRangeSlider = 7
       TextList = 8
       TextBox_Number = 9
       TextBox_String = 10
       Radio_Option = 11
       Button_Script = 12      ' Rendered as a button, executes a
when activated.
       Color_Picker = 13
```

ControlLocation = Specifies the desired location of the contro
row/column
ControlUse = If the pair is to control a specifc function, such as
On/Off/Dim, this specifies this function. This makes it easier to
basic devices without knowing the label. For example, some de
may use "On" for On, and others may use "Bright Full" for on. By
checking this property it is easy to find the control element for c
controls. The possible values are:

```
   Not_Specified = 0
   _On = 1
   _Off = 2
   _Dim = 3
   _On_Alternate = 4
   _Play = 5             ' media control devices
   _Pause = 6
```

	_Stop = 7 _Forward = 8 _Rewind = 9 _Repeat = 10 _Shuffle = 11 **ControlValue** = The value for this pair, use when controlling the device. **Range** = Specifies the range of this pair. This is used for pairs represent multiple values, such as the dim level of a light. A ligh specifiy a range of 1-99 for the dim levels. The range also spec the label using RangeStatusPrefix (such as "Dim"), and RangeStatusSuffix (such as "%"). The formatted status of the dev then be formatted as "Dim ## %".
/JSON?request=controldevicebyvalue&ref=###&value=#	Control a device given the device's reference number "ref", and "value". For example, if a light has a value of 0 for off, the follow would turn off the device with reference # 3570: /JSON?request=controldevicebyvalue&ref=3570&value=0 The return is the current JSON formatted status of the device (s return as "getstatus"), or the string "error".
/JSON?request=controldevicebylabel&ref=###&label=label	Control a device by label, this is the label as returned by the "getcontrol" parameter. For example, if the device has a label "(turn a device on, the following URL would turn it on: /JSON?request=controldevicebylabel&ref=3570&label=On The return is the current JSON formatted status of the device (s return as "getstatus"), or the string "error".
/JSON?request=getevents	Returns the names of all events in the system. An event is an ac be performed such as controlling a light, a sequence of lights, a thermostat, etc. Events have two properties, a group name and event name. This command returns the group name and event for all events. These two pieces of information are used to cont event. Example: `{ "Name":"HomeSeer Events", "Version":"1.0", "Events":[ { "Group":"Lighting", "Name":"Outside Lights Off" }, (MORE EVENTS FOLLOW)`
/JSON?request=runevent&group=GROUPNAME&name=EVENTNAME	This command will execute the actions of an event. Pass the gro name and event name. The group and name are not case sensi **Note:** Only available if A2Z-Link is enabled for automation.

/JSON?request=speak&phrase=text&host=HOST:NAME	Speaks the given phrase using text-to-speech. phrase = the phrase to speak host = the speaker host to speak out of. HomeSeer supports m hosts, like PC's and mobile devices. Each device is assgined a u host:name ID. For example, a host on the PC named "hometroll the name "Android" would have the host name: HomeTroller:An this is added to the host parameter, then the phrase will be spc that host only. Many hosts can be added and are separated by a comma, IE: host=HomeTroller:Android,iPhone:bill **Note:** Only available if A2Z-Link is enabled for automation.
/JSON?request=getlocations	Returns all the location names for location 1 and location 2
/JSON?request=getcounter&counter=NAME	Returns the value for the given named counter

/JSON?request=getsetting&setting=SETTING_NAME	Returns the value for a specific settting. For example, the settin
	name of location 1 is called "gLocLabel". To get the name of this
	use:
	
	/JSON?request=getsetting&setting=gLocLabel
	
	The return might be:
	
	`{`
	`"Value":"Room"`
	`}`

POST requests

JSON post requests can be used to control devices. The following request will turn a device on using the device value. In this example, a value of 255 will turn this device on:

Post data: {'action' : 'controlbyvalue', 'deviceref' : '3570', 'value' : '255'}
Url: ip_address/JSON

To control a device by label. In this case the label "On" will turn the device on:

Post data: {'action' : 'controlbylabel', 'deviceref' : '3570', 'label' : 'on'}

To run the actions of an event:

Post data: {'action' : 'runevent', 'group' : 'GROUPNAME', 'name' : 'EVENTNAME'}

See also

Index