Hunter Douglas PowerView® Hub API for Home Automation Integration

Appendix document

by

Jacob Laursen

and

Andrew Fiddian-Green

**Table of Contents**

[Overview 5](#_Toc95333721)

[Document Conventions 5](#_Toc95333722)

[Version information 5](#_Toc95333723)

[Acknowledgements 5](#_Toc95333724)

[Resources 5](#_Toc95333725)

[Repeaters 5](#_Toc95333726)

[Scene Collections 8](#_Toc95333727)

[Scheduled Events 12](#_Toc95333728)

[Shades 16](#_Toc95333729)

[Addenda to Main Specification 18](#_Toc95333730)

[Shade schema 18](#_Toc95333731)

[ShadePosition schema 19](#_Toc95333732)

[ShadeType schema 19](#_Toc95333733)

[Definitions 20](#_Toc95333734)

[BlinkEnabled 20](#_Toc95333735)

[EventType 20](#_Toc95333736)

[GroupId 20](#_Toc95333737)

[Name 20](#_Toc95333738)

[Repeater 20](#_Toc95333739)

[RepeaterFirmware 21](#_Toc95333740)

[RepeatersResponse 21](#_Toc95333741)

[SceneCollection 21](#_Toc95333742)

[SceneCollectionId 22](#_Toc95333743)

[SceneCollectionObject 22](#_Toc95333744)

[SceneCollectionsResponse 22](#_Toc95333745)

[ScheduledEvent 22](#_Toc95333746)

[ScheduledEventObject 23](#_Toc95333747)

[ScheduledEventsResponse 24](#_Toc95333748)

[ShadeAID 24](#_Toc95333749)

[ShadeBatteryKind 24](#_Toc95333750)

[ShadeCapabilities 24](#_Toc95333751)

[ShadeMotor 25](#_Toc95333752)

[ShadePosition 26](#_Toc95333753)

[ShadeType 27](#_Toc95333754)

[ShadeSignalStrength 28](#_Toc95333755)

[ShadeUpdate 28](#_Toc95333756)

[UniqueId 28](#_Toc95333757)

# Overview

This document is an appendix to the document **PowerView-Hub-REST-API-v2.pdf** which describes the PowerView® Hub REST API. The source of the original document is unknown, but it seems it might have been published by Hunter Douglas.

This appendix document describes some methods which are missing in the original document, or not fully described. This is partly based on reverse engineering by intercepting the communication between the PowerView app and the PowerView Hub.

## Document Conventions

See original document.

## Version information

*Version*: 1.0.5 (July 20th 2022)

## Acknowledgements

This document was created after some GitHub issue and pull request conversations between the two authors. Thanks for Andrew Fiddian-Green for providing detailed and well-formatted information about shade capabilities, shade positions, shade types and more.

# Resources

## Repeaters

Repeaters are known by the hub and it’s possible to see which shades are getting their signal forwarded through a particular repeater.

### Get all repeaters

GET /api/repeaters/

##### Description

* Gets a list of all repeater ids and the corresponding repeater data.
* The repeater data is returned in the same order as the repeater ids.
* If no repeaters exist, then empty arrays for repeater ids and repeater data are returned.

##### Responses

|  |  |  |
| --- | --- | --- |
| **HTTP**  **Code** | **Description** | **Schema** |
| **200** | Repeaters returned. | [RepeatersResponse](#_RepeatersResponse) |
| **400** | Bad client request. | No Content |
| **423** | Hub is temporarily busy for maintenance (Always returns UserData). | No Content |
| **500** | Internal server error. | No Content |

##### Example HTTP request

**Request path**

/api/repeaters

##### Example HTTP response

**Response 200**

{

"repeaterIds" : [ 7 ],

"repeaterData" : [ {

"id" : 7,

"blinkEnabled" : false,

"roomId" : 18,

"groupId" : 24,

"firmware" : {

"revision" : 2,

"subRevision" : 0,

"build" : 2928,

"index" : 33

},

"name" : "TmFtZQ=="

} ]

}

### Identify a repeater

GET /api/repeaters/{id}?identify=true

##### Description

* Identify a repeater by flashing LED.

##### Responses

|  |  |  |
| --- | --- | --- |
| **HTTP**  **Code** | **Description** | **Schema** |
| **200** | Repeater identified. | [RepeatersResponse](#_RepeatersResponse) |
| **400** | Bad client request. | No Content |
| **500** | Internal server error. | No Content |

##### Example HTTP request

**Request path**

/api/repeaters/17805?identify=true

### Update a repeater

PUT /api/repeaters/{id}

##### Description

* Update name and blinking enabled/disabled for a repeater.

##### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Name** | **Description** | **Schema** |
| **Path** | **id**  *required* | Unique id of the repeater. | Integer |

##### Responses

|  |  |  |
| --- | --- | --- |
| **HTTP**  **Code** | **Description** | **Schema** |
| **200** | Repeater identified. | [RepeatersResponse](#_RepeatersResponse) |
| **400** | Bad client request. | No Content |
| **500** | Internal server error. | No Content |

##### Example HTTP request

**Request path**

PUT /api/repeaters/17805

**Request body**

{

"repeater": {

"id": 16820,

"blinkEnabled": true,

"name": " UmVwZWF0ZXIgc292ZXbDpnJlbHNl"

}

}

## Scene Collections

Scene collections are no longer supported by the PowerView app. To be precise, creating new scene collections is no longer supported, but editing and deleting existing scene collections is still possible (for now). They are still fully supported by the PowerView Hub (for now).

### Get all scene collections

Please see original document. Important note, though: Hub v2 redirects /api/scenecollections to /api/sceneCollections. The latter can also be called directly to avoid redirection, but Hub v1 only supports the former:

GET /api/scenecollections

### Create a scene collection

POST /api/scenecollections/

##### Example HTTP request

**Request path**

POST /api/scenecollections/

**Request body**

{

"sceneCollection": {

"name": "QsO4cm4gaSBzZW5n",

"colorId": 12,

"iconId": 17,

"id": -1,

"order": 0,

"hkAssist": false

}

}

**Response 200**

{

"sceneCollection": {

"name": "QsO4cm4gaSBzZW5n",

"colorId": 12,

"iconId": 17,

"id": 16820,

"order": 0,

"hkAssist": false

}

}

##### An empty scene collection has now been created. The returned id can be used to include existing scenes in the scene collection.

### Modify a scene collection

PUT /api/scenecollections/{id}

##### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Name** | **Description** | **Schema** |
| **Path** | **id**  *required* | Unique id of the scene collection. | Integer |

##### Example HTTP request

**Request path**

PUT /api/scenecollections/16820

**Request body**

{

"sceneCollection": {

"name": "QsO4cm4gaSBzZW5n",

"colorId": 12,

"iconId": 17,

"id": 16820,

"order": 0,

"hkAssist": false

}

}

### Add a scene to a scene collection

POST /api/scenecollectionmembers/

##### Request body

{

"sceneCollectionMember": {

"sceneCollectionId": 16820,

"sceneId": 19165

}

}

##### At least two scenes are needed for the scene collection to make sense.

### Delete a scene from a scene collection

DELETE /api/scenecollectionmembers?sceneCollectionId={sceneCollectionId}&sceneId={sceneId}

##### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Name** | **Description** | **Schema** |
| **Path** | **sceneCollectionId**  *required* | Unique id of the scene collection. | Integer |
| **Path** | **sceneId**  *required* | Unique id of the scene. | Integer |

##### Responses

|  |  |  |
| --- | --- | --- |
| **HTTP**  **Code** | **Description** | **Schema** |
| **200** | Scene collection member deleted. |  |
| **400** | Bad client request. | No Content |
| **404** | Resource not found. | No Content |
| **423** | Hub is temporarily busy for maintenance (Always returns UserData). | No Content |
| **500** | Internal server error. | No Content |

##### Example HTTP request

**Request path**

DELETE /api/scenecollectionmembers?sceneCollectionId=16328&sceneId=19165

### Delete a scene collection

DELETE /api/scenecollections/{id}

##### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Name** | **Description** | **Schema** |
| **Path** | **id**  *required* | Unique id of the scene collection. | Integer |

##### Responses

|  |  |  |
| --- | --- | --- |
| **HTTP**  **Code** | **Description** | **Schema** |
| **204** | Scene collection deleted. | No Content |
| **400** | Bad client request. | No Content |
| **404** | Resource not found. | No Content |
| **423** | Hub is temporarily busy for maintenance (Always returns UserData). | No Content |
| **500** | Internal server error. | No Content |

##### Example HTTP request

**Request path**

DELETE /api/scenecollections/16820

## Scheduled Events

### Get all scheduled events

GET /api/scheduledevents/

##### Description

* Gets a list of all scheduled event ids and the corresponding scheduled event data.
* The scheduled event data is returned in the same order as the scheduled event ids.
* If no scheduled events exist, then empty arrays for scheduled event ids and scheduled event data are returned.

##### Responses

|  |  |  |
| --- | --- | --- |
| **HTTP**  **Code** | **Description** | **Schema** |
| **200** | Scheduled events returned. | [ScheduledEventsResponse](#_ScheduledEventsResponse) |
| **400** | Bad client request. | No Content |
| **423** | Hub is temporarily busy for maintenance (Always returns UserData). | No Content |
| **500** | Internal server error. | No Content |

##### Example HTTP request

**Request path**

/api/scheduledevents

##### Example HTTP response

**Response 200**

{

"scheduledEventIds": [

44234

],

"scheduledEventData": [

{

"enabled": true,

"sceneId": 7829,

"daySunday": true,

"dayMonday": true,

"dayTuesday": true,

"dayWednesday": true,

"dayThursday": true,

"dayFriday": true,

"daySaturday": true,

"eventType": 2,

"hour": 0,

"minute": 0,

"id": 44234

}

]

}

### Get a scheduled event

GET /api/scheduledevents/{id}

##### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Name** | **Description** | **Schema** |
| **Path** | **id**  *required* | Unique id of the scheduled event. | Integer |

##### Responses

|  |  |  |
| --- | --- | --- |
| **HTTP**  **Code** | **Description** | **Schema** |
| **200** | Scheduled event returned. | [ScheduledEventObject](#_ScheduledEventObject) |
| **400** | Bad client request. | No Content |
| **404** | Resource not found. | No Content |
| **423** | Hub is temporarily busy for maintenance (Always returns UserData). | No Content |
| **500** | Internal server error. | No Content |

##### Example HTTP request

**Request path**

/api/scheduledevent/44234

##### Example HTTP response

**Response 200**

{

"scheduledEvent": {

"enabled": true,

"sceneId": 7829,

"daySunday": true,

"dayMonday": true,

"dayTuesday": true,

"dayWednesday": true,

"dayThursday": true,

"dayFriday": true,

"daySaturday": true,

"eventType": 2,

"hour": 0,

"minute": 0,

"id": 44234

}

}

### Update a scheduled event

PUT /api/scheduledevents/{id}

##### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Name** | **Description** | **Schema** |
| **Path** | **id**  *required* | Unique id of the scheduled event. | Integer |

##### Responses

|  |  |  |
| --- | --- | --- |
| **HTTP**  **Code** | **Description** | **Schema** |
| **200** | Scheduled event updated. | [ScheduledEventObject](#_ScheduledEventObject) |
| **400** | Bad client request. | No Content |
| **404** | Resource not found. | No Content |
| **423** | Hub is temporarily busy for maintenance (Always returns UserData). | No Content |
| **500** | Internal server error. | No Content |

##### Example HTTP request

**Request path**

/api/scheduledevent/44234

##### Example HTTP response

**Request body**

{

"scheduledEvent": {

"enabled": false,

"sceneId": 7829,

"daySunday": true,

"dayMonday": true,

"dayTuesday": true,

"dayWednesday": true,

"dayThursday": true,

"dayFriday": true,

"daySaturday": true,

"eventType": 2,

"hour": 0,

"minute": 0,

"id": 44234

}

}

**Response 200**

{

"scheduledEvent": {

"enabled": false,

"sceneId": 7829,

"daySunday": true,

"dayMonday": true,

"dayTuesday": true,

"dayWednesday": true,

"dayThursday": true,

"dayFriday": true,

"daySaturday": true,

"eventType": 2,

"hour": 0,

"minute": 0,

"id": 44234

}

}

## Shades

### Get a shade

GET /api/shades/{id}

In addition to the parameters listed in the original document, this parameter exists:

##### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Name** | **Description** | **Schema** |
| **Path** | **id**  *required* | Unique id of the resource. | integer |
| **Query** | **survey**  *optional* | Request survey with signal strength information. | boolean |

##### Example HTTP request

**Request path**

/api/shades/7?survey=true

##### Example HTTP response

**Response 200**

{

"shade\_id" : 7,

"survey" : [

{

" neighbor\_id" : 2,

"rssi" : -84

}

]

}

### Update a shade

PUT /api/shades/{id}

##### Description

* See original document. Motion **calibrate** has been added to the [ShadeUpdate](#_ShadeUpdate) schema.
* Updates an already-existing shade.
* The object returned from the server contains the full representation of the updated shade (all fields, not just the updated ones)
* To perform a motion operation on a shade (down, up, jog, calibrate, …), then send a body that only has a motion operation in it.
* Only one of positions or motion may be updated.

##### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Name** | **Description** | **Schema** |
| **Path** | **id**  *required* | Unique id of the resource. | integer |

##### Body parameter

*Name* : body

*Flags* : required

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **shade**  *required* | **Example** : [ShadeUpdate](#_ShadeUpdate) | [ShadeUpdate](#_ShadeUpdate) |

# Addenda to Main Specification

The Hunter Douglas PowerView® Hub API for Home Automation Integration main specification describes some features and schemas in an incomplete manner, as described below.

## Shade schema

The Shade schema table on page 39 of the Hunter Douglas PowerView® Hub API for Home Automation Integration main specification is incomplete. The following table describes the missing elements.

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **aid**  *optional* | Alternate identifier for the shade? | [ShadeAID](#_ShadeAID) |
| **batteryKind**  *optional* | A description of the kind of battery used by the shade. E.g. (say) normal versus rechargeable battery. | [ShadeBatteryKind](#_ShadeBatteryKind) |
| **capabilities**  *optional* | A description of the capabilities of the shade. E.g. whether the shade is top down, bottom up, or side to side etc. | [ShadeCapabilities](#_ShadeCapabilities) |
| **motor**  *required* | A description of the type of motor in the shade. | [ShadeMotor](#_ShadeMotor) |
| **signalStrength**  *optional* | A number indicating the strength of the radio signal. | [ShadeSignalStrength](#_ShadeSignalStrength) |

## ShadePosition schema

The ShadePosdition schema on page 41 of the Hunter Douglas PowerView® Hub API for Home Automation Integration main specification is incomplete. Further information about the ShadePosition schema is provided in the Definitions chapter below.

## ShadeType schema

The ShadeType schema on page 43 of the Hunter Douglas PowerView® Hub API for Home Automation Integration main specification is empty. An explanation of the ShadeType schema is provided in the Definitions chapter below.

# Definitions

## 

## BlinkEnabled

Enable repeater blinking.

*Type* : boolean

## EventType

0 = Sunrise, 1 = Sunset, 2 = Time

*Type* : enum (0, 1, 2)

## GroupId

Unique id of the associated group.

*Type* : integer

## Name

Base64 encoded name.

*Type* : string (byte)

## Repeater

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **blinkEnabled**  *required* | **Example** : [BlinkEnabled](#_BlinkEnabled) | [BlinkEnabled](#_BlinkEnabled) |
| **groupId**  *required* | **Example** : [GroupId](#_GroupId) | [GroupId](#_GroupId) |
| **firmware**  *required* | **Example** : [RepeaterFirmware](#_RepeaterFirmware) | [RepeaterFirmware](#_RepeaterFirmware) |
| **id**  *required* | **Example** : [UniqueId](#_UniqueId_1) | [UniqueId](#_UniqueId_1) |
| **name**  *required* | **Example** : [Name](#BatteryStrength) | [Name](#BatteryStrength) |
| **roomId**  *required* | **Example** : [RoomId](#_bookmark55) | [RoomId](#_bookmark55) |

## RepeaterFirmware

*Polymorphism* : Composition

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **build**  *required* | Patch firmware version number.  **Minimum value** : 0 **Maximum value** : 65535 **Example** : 564 | integer |
| **index**  *required* | The index number.  **Example** : 25 | integer |
| **revision**  *required* | Major firmware version number.  **Minimum value** : 0 **Maximum value** : 255 **Example** : 2 | integer |
| **subRevision**  *required* | Minor firmware version number.  **Minimum value** : 0 **Maximum value** : 255 **Example** : 0 | integer |

## RepeatersResponse

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **repeaterData**  *required* | Repeater data for included repeaters.  **Example** : [ "[Repeater](#_bookmark69)" ] | < [Repeater](#_Repeater) > array |
| **repeaterIds**  *required* | Unique ids of all repeaters.  **Example** : [ "[UniqueId](#_UniqueId_1)" ] | < [UniqueId](#_UniqueId_1) > array |

## SceneCollection

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **colorId**  *required* | **Example** : [ColorId](#_bookmark24) | [ColorId](#_bookmark24) |
| **iconId**  *required* | **Example** : [IconId](#_bookmark45) | [IconId](#_bookmark45) |
| **id**  *required* | **Example** : [UniqueId](#_bookmark95) | [UniqueId](#_UniqueId) |
| **name**  *required* | **Example** : [Name](#_bookmark52) | [Name](#BatteryStrength) |
| **order**  *required* | **Example** : [Order](#_bookmark55) | [Order](#_bookmark55) |

## SceneCollectionId

The id of the Scene Collection to which this member belongs.

*Type* : integer

## SceneCollectionObject

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **sceneCollectio n**  *required* | **Example** : [SceneCollection](#NetworkNumber) | [SceneCollection](#NetworkNumber) |

## SceneCollectionsResponse

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **sceneCollectio nData**  *required* | Scene collection data for included scene collections.  **Example** : [ "[SceneCollection](#NetworkNumber)" ] | < [SceneCollection](#_SceneCollection) > array |
| **sceneCollectio nIds**  *required* | Unique ids of all scene collections.  **Example** : [ "[UniqueId](#_UniqueId)" ] | < [UniqueId](#_UniqueId) > array |

## ScheduledEvent

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **enabled**  *required* | **Example** : [true](#_bookmark24) | boolean |
| **sceneId**  *optional* | **Example** : [UniqueId](#_UniqueId) | [UniqueId](#_UniqueId) |
| **sceneCollectionId**  *optional* | **Example** : [UniqueId](#_UniqueId) | [UniqueId](#_UniqueId) |
| **daySunday**  *required* | **Example** : [true](#_bookmark24) | boolean |
| **dayMonday**  *required* | **Example** : [true](#_bookmark24) | boolean |
| **dayTuesday**  *required* | **Example** : [true](#_bookmark24) | boolean |
| **dayWednesday**  *required* | **Example** : [true](#_bookmark24) | boolean |
| **dayThursday**  *required* | **Example** : [true](#_bookmark24) | boolean |
| **dayFriday**  *required* | **Example** : [true](#_bookmark24) | boolean |
| **daySaturday**  *required* | **Example** : [true](#_bookmark24) | boolean |
| **eventType**  *required* | **Example:** [EventType](#_EventType) | [EventType](#_EventType) |
| **hour**  *required* | **Example** : [0](#_bookmark52) | integer |
| **minute**  *required* | **Example** : [15](#_bookmark55) | integer |
| **id**  *required* | **Example** : [UniqueId](#_UniqueId) | [UniqueId](#_UniqueId) |

## ScheduledEventObject

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **scheduledEvent**  *required* | **Example** : [ScheduledEvent](#_ScheduledEvent) | [ScheduledEvent](#_ScheduledEvent) |

## ScheduledEventsResponse

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **scheduledEventData**  *required* | Scheduled event data for included scheduled events.  **Example** : [ "[ScheduledEvent](#SceneObject)" ] | < [ScheduledEvent](#SceneObject) > array |
| **scheduledEventIds**  *required* | Unique ids of all scheduled events.  **Example** : [ "[UniqueId](#_bookmark95)" ] | < [UniqueId](#_UniqueId) > array |

## ShadeAID

An alternative ID for the shade. (??)

*Type*: integer.

## ShadeBatteryKind

A description of the type of battery used by the shade. E.g. normal or rechargeable battery.

*Type*: string.

## ShadeCapabilities

Different types of shades have different physical capabilities as described by their ShadeCapabilities value in the table below.

*Type*: integer.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Value | Description | Primary Rail | Secondary Rail | Tilt on Closed | Tilt Anywhere | Tilt 180° |
| 0 | Bottom Up | yes |  |  |  |  |
| 1 | Bottom Up, Tilt 90° | yes |  | yes |  |  |
| 2 | Bottom Up, Tilt 180° | yes |  |  | yes | yes |
| 3 | Vertical, Tilt 180° | yes |  |  | yes | yes |
| 4 | Vertical | yes |  |  |  |  |
| 5 | Tilt Only 180° |  |  |  | yes | yes |
| 6 | Top Down | reversed |  |  |  |  |
| 7 | Top Down, Bottom Up | yes | yes |  |  |  |
| 8 | Dual Overlapped | yes | overlapped |  |  |  |
| 9 | Dual Overlapped Tilt 90° | yes | overlapped | yes |  |  |

Examples of shades with different capabilities are shown below...

**Bottom Up** Shades with standard bottom-up operation such as Alustra Woven Textures, Roller, Screen & Banded Shades, Duette/Applause Standard, Design Studio shades, Solera, Vignette Modern Roman Shades Provenance Woven Wood Shades.

**Bottom Up Tilt 90°** Shades with Bottom-Up lift and 90° Tilt. Includes Pirouette, Silhouette, Silhouette A Deux Front Shade.

**Bottom Up Tilt 180°** Lift and Tilt Horizontal Blind (Not sold in USA at this time).

**Vertical** Vertically oriented shades with horizontal traverse operation only. Include Skyline Left Stack, Right Stack, Split Stack; Provenance Vertical Drapery Left Stack, Right Stack, Split Stack; Duette/Applause Vertiglide Left Stack, Right Stack.

**Vertical Tilt 180°** Vertically oriented shades with horizontal traverse operation plus 180° Tilt. Includes Luminette Left Stack, Right Stack and Split Stack.

**Tilt Only 180°** Products with tilt-only operation such as Parkland, EverWood, Modern Precious Metals and Palm Beach Shutters

**Top Down** Top-Down (only) operation includes Duette/Applause Top-Down.

**Top Down Bottom Up** Shades with Top-Down/Bottom-Up (TDBU) operation or stacking Duolite operation including Duette/Applause TDBU, Solera TDBU, Vignette TDBU, Provenance TDBU; Alustrao Woven Textureso Romans TDBU, Duette/Applause Duolite.

**Dual Overlapped** Shades with lift only Duolite operation eg. Vignette Duolite, Roller/Screen Duolite (not released).

**Dual Overlapped** Duolite lift operation plus 90° tilt operation. Includes: Silhouette Duolite.

**Tilt 90°**

## ShadeMotor

A description of the motor. E.g. as shown in the example below.

*Type*: complex.

{

"motor": {

"revision": 51,

"subRevision": 51,

"build": 11825

}

}

## ShadePosition

The ShadePosition schema is mentioned on page 41 of the Hunter Douglas PowerView® Hub API for Home Automation Integration main specification, but the following provides additional information concerning its usage.

The ShadePosition schema has two *“required”* elements (position1 and positionKind1) and two *“optional”* elements (position2 and positionKind2). The latter optional elements are only needed on specific types of shades, depending on the ShadeCapabilities (see above).

|  |  |  |
| --- | --- | --- |
| ShadeCapabilities | Uses Position2/ PositionKind2 | Function |
| 0 | no | When the shade is **not** closed, positionKind1 and position1 refer to the shade primary rail kind and position. And when it **is** closed, positionKind1 and position1 refer to the vane/tilt kind and position. |
| 1 | yes | One of the positionKindX and positionX pairs of values refers to the shade primary rail kind and position. And the other pair refers to the vane/tilt kind and position. |
| 2 | yes | As for ShadeCapabilities = 1 above. |
| 3 | no | As for ShadeCapabilities = 0 above. |
| 4 | yes | As for ShadeCapabilities = 1 above. |
| 5 | yes | The positionKind1 and position1 values refer to the vane/tilt kind and position. |
| 6 | no | The positionKind1 and position1 values refer to the shade primary rail kind and position. However, the coordinate system is the reverse of that for ShadeCapabilities = 0 above. |
| 7 | yes | The positionKind1 and position1 values refer to the shade primary rail kind and position. The positionKind2 and position2 values refer to the shade secondary rail kind and position. |
| 8 | no | As for ShadeCapabilities = 6 above. (??) |
| 9 | yes | As for ShadeCapabilities = 1 above. |

Notes:

1. On shades that use both positionKind1/position1 and positionKind2/position2 elements, the JSON elements can be written or read in either order, because the shade can determine the context from examining the respective positionKindX values.
2. On shades that use both positionKind1/position1 and positionKind2/position2 elements, both sets of elements must always be read and written. If only the “*required*” elements (position1 and positionKind1) are written, then the shade will exclude the second *“optional”* elements from any of its future responses. i.e., the secondary position will become indeterminate.
3. On shades which support vane/tilt functionality, then for shades with 90° vane/tilt, the positionX value range is 0 ... 32’767, whereas for shades with 180° vane/tilt, the range is 0 ... 65’535.

## ShadeType

The ShadeType schema is mentioned on page 43 of the Hunter Douglas PowerView® Hub API for Home Automation Integration main specification, but the following table provides missing information concerning the actual different types of shades.

Notes:

1. This table is probably not complete; it should be updated whenever existing but undocumented types are discovered, or new types introduced; in this case, please contact the author of this document to suggest an update.
2. The table shows a link to the respective ShadeCapabilities; however, the given ShadeCapabilities value is based on limited observations from a few users, so if errors are found please contact the author of this document to suggest an update.

*Type*: integer.

|  |  |  |
| --- | --- | --- |
| ShadeType | Description | *ShadeCapabilities* |
| 1 | Roller/Solar | *0* |
| 4 | Roman | *0* |
| 5 | *Not yet reported by users* | *0* |
| 6 | Duette | *0* |
| 7 | Top Down | *6* |
| 8 | Duette Top Down, Bottom Up | *7* |
| 9 | Duette DuoLite Top Down, Bottom Up | *7* |
| 18 | Pirouette | *1* |
| 23 | Silhouette | *1* |
| 31 | Vignette | *0* |
| 33 | Duette Architella | *7* |
| 38 | Silhouette Duolite | *9* |
| 42 | Roller Blind (M25T) | *0* |
| 43 | Facette | *1* |
| 44 | Twist | *1* |
| 47 | Pleated Top Down, Bottom Up | *7* |
| 49 | AC Roller | *0* |
| 51 | Venetian | *2* |
| 54 | Vertical Slats, Left Stack | *3* |
| 55 | Vertical Slats, Right Stack | *3* |
| 56 | Vertical Slats, Split Stack | *3* |
| 62 | Venetian | *2* |
| 65 | Vignette Duolite | *8* |
| 66 | Shutter | *5* |
| 69 | Curtain, Left Stack | *4* |
| 70 | Curtain, Right Stack | *4* |
| 71 | Curtain, Split Stack | *4* |
| 79 | Duolite Lift | *8* |

## ShadeSignalStrength

The strength of the radio signal between the shade and the hub.

*Type*: integer.

## ShadeUpdate

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Schema** |
| **motion**  *optional* | The motion operation to perform on a shade or group.  **Example** : "jog" | enum (down, heart, jog, leftTilt, rightTilt, stop, up, calibrate) |

## UniqueId

Unique resource identifier.

*Type* : integer