

digitalSTROM-Server JSON

digitalSTROM

Version: v1.7*

July 5, 2022

*Revision: 2f262f0657d98f4af95a64a1b976f8c9167326fc

©2017 digitalSTROM AG. All rights reserved.

The digitalSTROM logo is a trademark of the digitalSTROM. Use of this logo for commercial purposes without the prior written consent of digitalSTROM may constitute trademark infringement and unfair competition in violation of international laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. digitalSTROM retains all intellectual property rights associated with the technology described in this document. This document is intended to assist developers to develop applications that use or integrate digitalSTROM technologies.

Every effort has been made to ensure that the information in this document is accurate. digitalSTROM is not responsible for typographical errors.

digitalSTROM AG
Building Technology Park Zürich
Brandstrasse 33
CH-8952 Schlieren
Switzerland

Even though digitalSTROM has reviewed this document, digitalSTROM MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT THIS DOCUMENT IS PROVIDED "AS IS", AND YOU, THE READER ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL DIGITALSTROM BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. NO DIGITALSTROM AGENT OR EMPLOYEE IS AUTHORIZED TO MAKE ANY MODIFICATION, EXTENSION, OR ADDITION TO THIS WARRANTY.

Contents

1	Introduction	9
2	Apartment	10
2.1	Name	10
2.1.1	getName	10
2.1.2	setName	10
2.2	Scene	11
2.2.1	callScene	11
2.2.2	saveScene	11
2.2.3	undoScene	12
2.2.4	getLockedScenes	12
2.3	Value	13
2.3.1	Set Device Output Value	13
2.3.2	Get Binary Input Information	14
2.4	Groups	14
2.4.1	getReachableGroups	15
2.5	Structure	15
2.5.1	getStructure	16
2.5.2	getDevices	22
2.5.3	getCircuits	24
2.5.4	getVdcs	25
2.5.5	removeMeter	26
2.6	Sensors	26
2.6.1	Get Assigned Sensors	26
2.6.2	Get Sensor Values	27
2.6.3	Set Weather Information	30
2.7	Heating	30
2.7.1	Get Temperature Control Status	30
2.7.2	Get Temperature Control Configuration	31
2.7.3	Get Temperature Control Values	33
2.7.4	Get Temperature Control Configuration v2	34
2.8	DevicesFirstSeen	35
2.8.1	setDevicesFirstSeen	35
2.9	ModelFeatures	36
2.9.1	getModelFeatures	36
2.10	MuteFire	37
2.11	Controller Statuses	37
2.11.1	getControllerStatus	37
3	Zone	39
3.1	Common	39
3.2	Action	39
3.2.1	getActions	39
3.2.2	callAction	41
3.3	Name	42
3.3.1	getName	42
3.3.2	setName	42
3.4	Scene	43
3.4.1	callScene	43
3.4.2	saveScene	43
3.4.3	undoScene	44
3.4.4	sceneGetName	44
3.4.5	sceneSetName	45

3.4.6	getReachableScenes	46
3.4.7	getLastCalledScene	47
3.5	Value	47
3.5.1	Set Output Value	47
3.5.2	Blink	48
3.5.3	Send Sensor Value	48
3.5.4	Set Status Field	49
3.6	Sensors	50
3.6.1	Set Sensor Source	50
3.6.2	Clear Sensor Source	50
3.6.3	Get Assigned Sensors	51
3.6.4	Get Sensor Values	52
3.7	Heating	52
3.7.1	Get Temperature Control Status	52
3.7.2	Get Temperature Control Configuration	53
3.7.3	Set Temperature Control Configuration	55
3.7.4	Get Temperature Control Values	56
3.7.5	Set Temperature Control Values	57
3.7.6	Get Temperature Control Configuration v2	57
3.7.7	Set Temperature Control Configuration v2	59
3.7.8	Set Temperature Control State	60
3.7.9	Get Temperature Control Internals	60
4	Device	62
4.1	Common	62
4.2	Name	62
4.2.1	getName	62
4.2.2	setName	63
4.2.3	getSpec	63
4.3	First seen	64
4.3.1	getFirstSeen	64
4.4	Groups	64
4.4.1	getGroups	64
4.5	Scene	65
4.5.1	callScene	65
4.5.2	saveScene	65
4.5.3	undoScene	66
4.5.4	turnOn	66
4.5.5	turnOff	67
4.5.6	increaseValue	67
4.5.7	decreaseValue	67
4.6	Value	68
4.6.1	Set Value	68
4.6.2	Set Output Value	69
4.6.3	Get Output Value	69
4.6.4	Get Scene Value	70
4.6.5	Set Scene Value	71
4.6.6	Get Scene Mode	72
4.6.7	Set Scene Mode	72
4.6.8	Blink	73
4.6.9	Get Output Channel Value	73
4.6.10	Set Output Channel Value	75
4.6.11	Get Output Channel Value v2	75
4.6.12	Set Output Channel Value v2	76
4.6.13	Get Output Channel Scene Value	77

4.6.14	Set Output Channel Scene Value	78
4.6.15	Get Output Channel Scene Value v2	78
4.6.16	Set Output Channel Scene Value v2	79
4.6.17	Get Output Channel Don't Care Flags	80
4.6.18	Set Output Channel Don't Care Flag	81
4.7	Configuration	81
4.7.1	setButtonID	81
4.7.2	setButtonInputMode	82
4.7.3	setOutputMode	82
4.7.4	setJokerGroup	83
4.7.5	setButtonActiveGroup	83
4.7.6	getTransitionTime	84
4.7.7	setTransitionTime	84
4.7.8	setConfig	85
4.7.9	getConfig	86
4.7.10	getConfigWord	86
4.7.11	setCardinalDirection	87
4.7.12	getCardinalDirection	88
4.7.13	setWindProtectionClass	88
4.7.14	getWindProtectionClass	89
4.7.15	setFloor	89
4.7.16	getFloor	90
4.7.17	getMaxMotionTime	90
4.7.18	setMaxMotionTime	91
4.7.19	getOutputAfterImpulse	91
4.7.20	setOutputAfterImpulse	92
4.7.21	setVisibility	92
4.7.22	setSupportedBasicScenes	93
4.7.23	getSupportedBasicScenes	93
4.7.24	setIgnoreOperationLock	94
4.7.25	getIgnoreOperationLock	94
4.8	Sensor	95
4.8.1	Get Sensor Value	95
4.8.2	Get Sensor Type	95
4.8.3	getSensorEventTableEntry	96
4.8.4	setSensorEventTableEntry	97
4.9	Programming	97
4.9.1	Set Programming Mode	97
4.9.2	Add To Area	98
4.9.3	Remove From Area	98
4.9.4	Get Transmission Quality	99
4.10	Heating and valve actuators	99
4.10.1	setHeatingGroup	99
4.10.2	getValvePwmState	100
4.10.3	getValvePwmMode	100
4.10.4	setValvePwmMode	101
4.10.5	getValveControlMode	102
4.10.6	setValveControlMode	102
4.10.7	getValveTimerMode	103
4.10.8	setValveTimerMode	104
4.11	Single Device Info	104
4.11.1	Get Info Static	104
4.11.2	Get Info Operational	111
4.11.3	Get Info Custom	113
4.11.4	Get Info	114

4.11.5	Set Property	114
4.11.6	Set Custom Action	115
4.11.7	Call Action	115
4.11.8	Get Apartment Scenes	116
4.12	Test presence	117
4.12.1	testPresence	117
4.13	Button Usage	117
4.13.1	getButtonUsage	117
4.13.2	setButtonUsage	118
5	Circuit	119
5.1	Common	119
5.2	Name	119
5.2.1	getName	119
5.2.2	setName	119
5.3	Energy Meter	120
5.3.1	getConsumption	120
5.3.2	getEnergyMeterValue	120
5.3.3	getMeteringConfiguration	121
5.3.4	setMeteringConfiguration	121
5.4	Configuration	122
5.4.1	learnIn	122
5.4.2	learnOut	122
5.4.3	firmwareCheck	123
5.4.4	firmwareUpdate	123
5.4.5	storeAccessToken	124
5.5	Reregister devices	124
5.5.1	reregisterDevices	124
6	Structure	126
6.1	Zone	126
6.1.1	addZone	126
6.1.2	removeZone	126
6.1.3	floorAddZone	127
6.2	Floor	127
6.2.1	addFloor	127
6.2.2	removeFloor	128
6.2.3	setFloorName	128
6.3	Group	129
6.3.1	addGroup	129
6.3.2	removeGroup	130
6.3.3	groupSetName	130
6.3.4	groupSetColor	131
6.3.5	groupSetConfiguration	131
6.3.6	groupGetConfiguration	132
6.4	Cluster	133
6.4.1	addCluster	133
6.4.2	removeCluster	133
6.4.3	clusterSetName	134
6.4.4	clusterSetColor	134
6.4.5	clusterSetConfigLock	135
6.5	Device	135
6.5.1	zoneAddDevice	135
6.5.2	removeDevice	136
6.5.3	groupAddDevice	137

6.5.4	groupRemoveDevice	138
6.5.5	clusterAddDevice	139
6.5.6	clusterRemoveDevice	140
7	Event and State	142
7.1	Raise Event	142
7.1.1	raise	142
7.2	Event Subscription	142
7.2.1	subscribe	142
7.2.2	unsubscribe	143
7.2.3	get	144
7.3	State	144
7.3.1	set	144
8	Metering	146
8.1	Metering	146
8.1.1	getResolution	146
8.1.2	getSeries	146
8.1.3	getValues	147
8.1.4	getAggregatedValues	149
8.1.5	getLatest	150
8.1.6	getAggregatedLatest	151
9	System	153
9.1	System Information	153
9.1.1	version	153
9.1.2	time	153
9.1.3	getDSID	154
9.2	Authentication	154
9.2.1	login	154
9.2.2	logout	155
9.2.3	loggedInUser	155
9.2.4	requestApplicationToken	156
9.2.5	enableToken	156
9.2.6	revokeToken	157
9.2.7	loginApplication	157
9.3	Logging	158
9.3.1	logChannelSeverities	158
9.3.2	logChannelSeverities/[id]	158
9.3.3	logChannelSeverities/[id]?method=put	159
10	Property Tree	160
10.1	Basic Property Tree Operations	160
10.1.1	getString	160
10.1.2	setString	160
10.1.3	getInteger	161
10.1.4	setInteger	161
10.1.5	getBoolean	161
10.1.6	setBoolean	162
10.1.7	getChildren	162
10.1.8	getType	163
10.1.9	getFlags	163
10.1.10	setFlag	164
10.1.11	remove	164
10.2	Property Query	165

10.2.1 query	165
10.2.2 query2	166
11 Database	168
11.1 Database Query	168
11.1.1 query	168

1 Introduction

All requests are sent using HTTP GET and parameters added to the query string url like:

```
/json/apartment/setName?name="MyDigitalStromServer"&username=dssadmin&password=secret
```

If not properly authenticated the HTTP Status 403 is returned and the error response contains:

```
{  
  "ok": false,  
  "message": "not logged in"  
}
```

If an unknown method is requested the error message "Unhandled Function" is returned:

```
{  
  "ok": false,  
  "message": "Unhandled function"  
}
```

If a request has been successfully processed the JSON answer contains an "ok" and an optional "result" field. The result array is explained in the particular sections.

ok	true
result	array of result values

Where Group Names are allowed the following table lists the possible names.

Name	Group Id	Description
yellow	1	Light
gray	2	Light
blue	3	Climate

2 Apartment

2.1 Name

2.1.1 getName

Returns the user defined name of the installation.

Synopsis

HTTP GET /json/apartment/getName

Parameter

None

Response

HTTP Status 200

name	identifier string for the installation
------	----------------------------------------

Sample

```
GET /json/apartment/getName
{
  "ok":true,
  "result" :
  {
    "name" : "digitalStromInstallationHansMustermann"
  }
}
```

2.1.2 setName

Sets the installation name.

Synopsis

HTTP GET /json/apartment/setName

Parameter	Description	Remarks
newName	identifier string for the installation	Mandatory

Parameter

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/apartment/setName?newName="MyDdSS"
{
  "ok":true
}
```

2.2 Scene

2.2.1 callScene

Excutes the scene sceneNumber on a group of devices.

Synopsis

HTTP GET /json/apartment/callScene

Parameter

Parameter	Description	Remarks
sceneNumber	Numerical value	Mandatory
groupID	Number of the target group	Optional
groupName	Name of the target group	Optional
force	Boolean value, if set a forced scene call is issued	Optional

If the group parameters are omitted the command is sent as broadcast to all zones and all devices.

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/apartment/callScene?sceneNumber=65
{
  "ok":true
}
```

2.2.2 saveScene

Tells devices to store their current output values as a default for the scene sceneNumber.

Synopsis

HTTP GET /json/apartment/saveScene

Parameter

Parameter	Description	Remarks
sceneNumber	Numerical value	Mandatory
groupID	Number of the target group	Optional
groupName	Name of the target group	Optional

If the group parameters are omitted the command is sent as broadcast to all zones and all devices.

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/apartment/saveScene?sceneNumber=65
{
  "ok":true
}
```

2.2.3 undoScene

Tells devices to restore their output values to the previous state if the current scene matches the sceneNumber.

Synopsis

HTTP GET /json/apartment/undoScene

Parameter

Parameter	Description	Remarks
sceneNumber	Numerical value	Mandatory
groupID	Number of the target group	Optional
groupName	Name of the target group	Optional

If the group parameters are omitted the command is sent as broadcast to all zones and all devices.

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/apartment/undoScene?sceneNumber=65
{
  "ok":true
}
```

2.2.4 getLockedScenes

Retrieves scene numbers of scenes that are currently locked because of an update of device scene tables.

Synopsis

HTTP GET /json/apartment/getLockedScenes

Parameter None

Response

HTTP Status 200

result.lockedScenes[] array of scene numbers that are currently locked

Sample

```
GET /json/apartment/getLockedScenes
{
  "ok" : true,
  "result" :
  {
    "lockedScenes" : []
  }
}
```

2.3 Value

2.3.1 Set Device Output Value

Set the output value of a group of devices to a given value.

Notice Setting output values directly bypasses the group state machine and is unrecommended.

Synopsis

HTTP GET /json/apartment/setValue

Parameter

Parameter	Description	Remarks
value	Numerical value	Mandatory
groupID	Number of the target group	Optional
groupName	Name of the target group	Optional

If the group parameters are omitted the command is sent as broadcast to all devices.

Notice Setting output values without a group identification is strongly unrecommended.

Response

HTTP Status 200

ok	true
data	array of devices that have binary inputs

Sample

```
GET /json/apartment/setValue?value=0&groupID=2
{
  "ok":true,
}
```

2.3.2 Get Binary Input Information

Retrieve the information about binary inputs of all devices.

Synopsis

HTTP GET /json/apartment/getDeviceBinaryInputs

Parameter

None

Response

HTTP Status 200

ok	true
devices	array of devices that have binary inputs

Sample

```
GET /json/apartment/getDeviceBinaryInputs
{
  "result":
  {
    "devices":
    [
      {
        "dsuid": "3504175fe00000000000000000d91100",
        "binaryInputs":
        [
          {
            "targetGroupType": 0,
            "targetGroup": 8,
            "inputType": 11,
            "inputId": 15,
            "state": 1
          }
        ]
      },
      {
        "dsuid": "3504175fe00000000000000000d91000",
        "binaryInputs":
        [
          {
            "targetGroupType": 0,
            "targetGroup": 16,
            "inputType": 8,
            "inputId": 15,
            "state": 1
          }
        ]
      }
    ]
  },
  "ok": true
}
```

2.4 Groups

2.4.1 getReachableGroups

Returns a list of groups for which are actuators actually present in the installation.

Synopsis

HTTP GET /json/apartment/getReachableGroups

Parameter

None

Response

HTTP Status 200

result.zones	array of zones in the installation
result.zones[].groups	array of groups in a zone

Sample

```
GET /json/apartment/getReachableGroups
```

```
{
  "ok": true,
  "result": {
    "zones": [
      {
        "zoneID": 0,
        "name": "",
        "groups": [
          64,
          69
        ]
      },
      {
        "zoneID": 1223,
        "name": "Wohnen",
        "groups": [
          1,
          2,
          7
        ]
      },
      {
        "zoneID": 1241,
        "name": "Schlafen",
        "groups": [
          1,
          5,
          7
        ]
      },
      {
        "zoneID": 1237,
        "name": "Essen",
        "groups": [
          1,
          6
        ]
      }
    ]
  }
}
```

2.5 Structure

2.5.1 getStructure

Returns an object containing the structure of the apartment. This includes detailed information about all zones, groups and devices.

Synopsis

HTTP GET /json/apartment/getStructure

Parameter

None

Response

HTTP Status 200

result.apartment.zones	array of zone information
result.apartment.zones[].devices	array of device information in each zone
result.apartment.zones[].devices[].groups	group membership of each device groups
result.apartment.zones[].groups	array of group information in each zone
result.apartment.zones[].groups[].devices	array of devices per group in a zone
result.apartment.zones[].groups[].devices[x].modelFeatures	object of device specific model features. These model features have the same format as returned from the getModelFeatures call (see 2.9) and override the features given there for this specific device. This node is optional.
result.apartment.clusters	array of cluster information
result.apartment.clusters[].devices[]	cluster membership of each device clusters
result.apartment.floors	array of floor information

Sample

```
GET /json/apartment/getStructure
{
  "ok": true,
  "result": {
    "apartment": {
      "zones": [
        {
          "id": 0,
          "name": "",
          "isPresent": false,
          "floor": 0,
          "devices": [
            {
              "dSUID": "3504175fe0000000000000000000182f600",
              "name": "Regalleuchte",
              "functionID": 4152,
              "productRevision": 49955,
              "productID": 6344,
              "hwInfo": "GE-SDS200",
            }
          ]
        }
      ]
    }
  }
}
```



```

        "3504175fe0000000000000000000439c00",
        "3504175fe000000000000000000043a700"
    ]
},
{
    "id": 1,
    "name": "yellow",
    "isPresent": true,
    "devices": [
        "3504175fe0000000000000000000182f600",
        "3504175fe00000000000000000000439c00",
        "3504175fe0000000000000000000043a700"
    ]
},
{
    "id": 2,
    "name": "gray",
    "isPresent": true,
    "devices": []
},
{
    "id": 3,
    "name": "blue",
    "isPresent": true,
    "devices": []
},
{
    "id": 4,
    "name": "cyan",
    "isPresent": true,
    "devices": []
},
{
    "id": 5,
    "name": "magenta",
    "isPresent": true,
    "devices": [
        "3504175fe0000000000000000000151fd00"
    ]
},
{
    "id": 6,
    "name": "red",
    "isPresent": true,
    "devices": [
        "3504175fe000000000000000000042dc00"
    ]
},
{
    "id": 7,
    "name": "green",
    "isPresent": true,
    "devices": []
},
{
    "id": 8,
    "name": "black",
    "isPresent": true,
    "devices": [
        "3504175fe0000000000000000000151fd00"
    ]
}
],
{
    "id": 989,
    "name": "Wohnen",
    "isPresent": true,
    "floor": 1,
    "devices": [
        {
            "dSUID": "3504175fe0000000000000000000182f600",
            "name": "Regalleuchte",
            "functionID": 4152,
            "productRevision": 49955,
            "productID": 6344,
            "hwInfo": "GE-SDS200",
            "meterDSUID": "3504175fe00000000000010000003dd00",

```

```

    "busID": 97,
    "zoneID": 989,
    "isPresent": false,
    "lastDiscovered": "2012-10-24 11:17:29",
    "firstSeen": "2012-10-22 16:22:02",
    "inactiveSince": "2012-10-22 16:22:02",
    "outputMode": 22,
    "buttonID": 0,
    "buttonActiveGroup": 1,
    "buttonInputMode": 0,
    "buttonInputIndex": 0,
    "buttonInputCount": 1,
    "groups": [
      "1"
    ],
    "modelFeatures": {
      "dontcare": true,
      "blink": false,
      "ledauto": true
    }
  },
  {
    "dSUID": "3504175fe00000000000000000000000439c00",
    "name": "Stehlampe",
    "functionID": 4152,
    "productRevision": 789,
    "productID": 200,
    "hwInfo": "GE-KM200",
    "meterDSUID": "3504175fe000000000000000000000003dd00",
    "busID": 153,
    "zoneID": 989,
    "isPresent": false,
    "lastDiscovered": "2012-10-24 11:17:29",
    "firstSeen": "2012-10-22 16:22:02",
    "inactiveSince": "2012-10-22 16:22:02",
    "outputMode": 22,
    "buttonID": 0,
    "buttonActiveGroup": 1,
    "buttonInputMode": 0,
    "buttonInputIndex": 0,
    "buttonInputCount": 1,
    "groups": [
      "1"
    ]
  }
],
"groups": [
  {
    "id": 0,
    "name": "broadcast",
    "isPresent": false,
    "devices": [
      "3504175fe00000000000000000000000182f600",
      "3504175fe00000000000000000000000439c00"
    ]
  },
  {
    "id": 1,
    "name": "yellow",
    "isPresent": true,
    "devices": [
      "3504175fe00000000000000000000000182f600",
      "3504175fe00000000000000000000000439c00"
    ]
  },
  {
    "id": 2,
    "name": "gray",
    "isPresent": true,
    "devices": []
  },
  {
    "id": 3,
    "name": "blue",
    "isPresent": true,
    "devices": []
  }
],
{

```

```

        "id": 4,
        "name": "cyan",
        "isPresent": true,
        "devices": []
    },
    {
        "id": 5,
        "name": "magenta",
        "isPresent": true,
        "devices": [
            "3504175fe000000000000000000015fd00"
        ]
    },
    {
        "id": 6,
        "name": "red",
        "isPresent": true,
        "devices": [
            "3504175fe000000000000000000042dc00"
        ]
    },
    {
        "id": 7,
        "name": "green",
        "isPresent": true,
        "devices": []
    },
    {
        "id": 8,
        "name": "black",
        "isPresent": true,
        "devices": [
            "3504175fe000000000000000000015fd00"
        ]
    }
]
},
{
    "id": 1038,
    "name": "Schlafen",
    "isPresent": true,
    "floor": 1,
    "devices": [
        {
            "dSUID": "3504175fe000000000000000000043a700",
            "name": "Deckenlicht",
            "functionID": 4152,
            "productRevision": 789,
            "productID": 200,
            "hwInfo": "GE-KM200",
            "meterDSUID": "3504175fe000000000000010000003dd00",
            "busID": 784,
            "zoneID": 1038,
            "isPresent": true,
            "lastDiscovered": "2012-10-26T15:36:30",
            "firstSeen": "2012-10-22T16:22:02",
            "inactiveSince": "1970-01-01T01:00:00",
            "outputMode": 22,
            "buttonID": 5,
            "buttonActiveGroup": 1,
            "buttonInputMode": 0,
            "buttonInputIndex": 0,
            "buttonInputCount": 1,
            "groups": [
                "1"
            ]
        }
    ],
    "groups": [
        {
            "id": 0,
            "name": "broadcast",
            "isPresent": true,
            "devices": [
                "3504175fe000000000000000000043a700"
            ]
        }
    ],
    {

```

```

        "id": 1,
        "name": "yellow",
        "isPresent": true,
        "devices": [
            "3504175fe0000000000000000043a700"
        ]
    },
    {
        "id": 2,
        "name": "gray",
        "isPresent": true,
        "devices": []
    },
    {
        "id": 3,
        "name": "blue",
        "isPresent": true,
        "devices": []
    },
    {
        "id": 4,
        "name": "cyan",
        "isPresent": true,
        "devices": []
    },
    {
        "id": 5,
        "name": "magenta",
        "isPresent": true,
        "devices": []
    },
    {
        "id": 6,
        "name": "red",
        "isPresent": true,
        "devices": []
    },
    {
        "id": 7,
        "name": "green",
        "isPresent": true,
        "devices": []
    },
    {
        "id": 8,
        "name": "black",
        "isPresent": true,
        "devices": []
    }
    ]
}
],
"clusters": [
    {
        "id": 16,
        "name": "South-Class10-07",
        "color": 2,
        "applicationType": 2,
        "isPresent": true,
        "isValid": true,
        "CardinalDirection": "south",
        "ProtectionClass": 1,
        "isAutomatic": true,
        "configurationLock": false,
        "devices": [
            "302ed89f43f000000000d000009f75a00",
            "303505d7f800000000000f40000aa96b00"
        ],
        "lockEvents": []
    },
    {
        "id": 17,
        "name": "Markisen",
        "color": 2,
        "applicationType": 2,
        "isPresent": true,
        "isValid": true,
        "CardinalDirection": "none",

```



```

    ]
  },
  {
    "dSUID": "3504175fe000000000000000000439c00",
    "name": "Stehlampe",
    "functionID": 4152,
    "productRevision": 789,
    "productID": 200,
    "hwInfo": "GE-KM200",
    "meterDSUID": "3504175fe00000000000010000003dd00",
    "busID": 153,
    "zoneID": 989,
    "isPresent": false,
    "lastDiscovered": "2012-10-24T11:17:29",
    "firstSeen": "2012-10-22T16:22:02",
    "inactiveSince": "2012-10-22T16:22:02",
    "outputMode": 22,
    "buttonID": 0,
    "buttonActiveGroup": 1,
    "buttonInputMode": 0,
    "buttonInputIndex": 0,
    "buttonInputCount": 1,
    "groups": [
      "1"
    ]
  },
  {
    "dSUID": "3504175fe00000000000000000151fd00",
    "name": "Fernseher",
    "functionID": 33041,
    "productRevision": 41761,
    "productID": 5320,
    "hwInfo": "SW-ZWS200",
    "meterDSUID": "3504175fe00000000000010000003dd00",
    "busID": 693,
    "zoneID": 989,
    "isPresent": false,
    "lastDiscovered": "2012-10-24T11:17:29",
    "firstSeen": "2012-10-22T16:22:02",
    "inactiveSince": "2012-10-22T16:22:02",
    "outputMode": 39,
    "buttonID": 0,
    "buttonActiveGroup": 5,
    "buttonInputMode": 0,
    "buttonInputIndex": 0,
    "buttonInputCount": 1,
    "groups": [
      "5",
      "8"
    ]
  },
  {
    "dSUID": "3504175fe00000000000000000000439c00",
    "name": "Wandlampe",
    "functionID": 4144,
    "productRevision": 789,
    "productID": 1234,
    "hwInfo": "GE-TKM210",
    "meterDSUID": "3504175fe00000000000010000003dd00",
    "busID": 782,
    "zoneID": 989,
    "isPresent": false,
    "lastDiscovered": "2012-10-24T11:17:29",
    "firstSeen": "2012-10-22T16:22:02",
    "inactiveSince": "2012-10-22T16:22:02",
    "outputMode": 22,
    "buttonID": 4,
    "buttonActiveGroup": 1,
    "buttonInputMode": 0,
    "buttonInputIndex": 0,
    "buttonInputCount": 1,
    "groups": [
      "1"
    ]
  },
  {
    "dSUID": "3504175fe000000000000000000043a700",
    "name": "Deckenlicht",

```

```

    "functionID": 4152,
    "productRevision": 789,
    "productID": 200,
    "hwInfo": "GE-KM200",
    "meterDSUID": "3504175fe00000000000010000003dd00",
    "busID": 784,
    "zoneID": 1038,
    "isPresent": true,
    "lastDiscovered": "2012-10-26T15:36:30",
    "firstSeen": "2012-10-22T16:22:02",
    "inactiveSince": "1970-01-01T01:00:00",
    "outputMode": 22,
    "buttonID": 5,
    "buttonActiveGroup": 1,
    "buttonInputMode": 0,
    "buttonInputIndex": 0,
    "buttonInputCount": 1,
    "groups": [
      "1"
    ]
  },
  {
    "dSUID": "3504175fe0000000000000000042dc00",
    "name": "Paniktaster",
    "functionID": 24896,
    "productRevision": 790,
    "productID": 1225,
    "hwInfo": "RT-TKM201",
    "meterDSUID": "3504175fe00000000000010000003dd00",
    "busID": 785,
    "zoneID": 989,
    "isPresent": false,
    "lastDiscovered": "2012-10-24T11:17:29",
    "firstSeen": "2012-10-23T16:23:38",
    "inactiveSince": "2012-10-24T11:01:40",
    "outputMode": 0,
    "buttonID": 17,
    "buttonActiveGroup": 154,
    "buttonInputMode": 20,
    "buttonInputIndex": 0,
    "buttonInputCount": 0,
    "groups": [
      "6"
    ]
  }
]
}

```

2.5.3 getCircuits

Returns an array containing all digitalSTROM-Meters of the apartment.

Synopsis

HTTP GET /json/apartment/getCircuits

Parameter

None

Response

HTTP Status 200

```
result.circuits array of digitalSTROM Meters
```

Sample

```
GET /json/apartment/getCircuits
{
```



```

"ok": true,
"result": {
  "circuits": [
    {
      "name": "dSM03DD-#1",
      "dSUID": "3504175fe0000000000000000000000016c4f00",
      "hwVersion": 721409,
      "armSwVersion": 17498112,
      "dspSwVersion": 16908800,
      "apiVersion": 517,
      "hwName": "",
      "isPresent": true,
      "isValid": true
    },
    {
      "name": "dSM040E-#2",
      "dSUID": "3504175fe0000000000000000000000016c4f00",
      "hwVersion": 721409,
      "armSwVersion": 17498112,
      "dspSwVersion": 16908800,
      "apiVersion": 517,
      "hwName": "",
      "isPresent": true,
      "isValid": true
    }
  ]
}
}

```

2.5.4 getVdcs

Returns an array containing all vDCs matching the given implementationId in the apartment.

Synopsis

HTTP GET /json/apartment/getVdcs

Parameter

Parameter	Description	Remarks
implementationId	implementationId of the vDC	Mandatory

Response

HTTP Status 200

```
result.vdcs array of vDCs
```

Sample

```

GET /json/apartment/getVdcs&implementationId=DALI_Device_Container
{
  "ok": true,
  "result": {
    "vdcs": [
      {
        "name": "DALI",
        "dSUID": "8b7dd1d5f2a158b780010dc3dd7f2a4a00",
        "DisplayID": "8...b7dd1d5",
        "hwVersion": 0,
        "hwVersionString": "",
        "swVersion": "1.6.0.9",
        "armSwVersion": 0,

```

```

        "dspSwVersion": 0,
        "apiVersion": 771,
        "hwName": "P44-DSB-DEHDDALI",
        "isPresent": true,
        "isValid": true,
        "busMemberType": 33,
        "hasDevices": true,
        "hasMetering": false,
        "VdcConfigURL": "http://172.17.0.77:80",
        "VdcModelUID": "1A07F24C6D7758CE8055ADDD65E5087300",
        "VdcHardwareGuid": "",
        "VdcHardwareModelGuid": "",
        "VdcImplementationId": "DALI_Device_Container",
        "VdcVendorGuid": "",
        "VdcOemGuid": "",
        "ignoreActionsFromNewDevices": false
    }
  ]
}

```

2.5.5 removeMeter

Removes an inactive digitalSTROM-Meter object from the installation.

Synopsis

HTTP GET /json/apartment/removeMeter

Parameter

Parameter	Description	Remarks
dSUID	dSUID of the digitalSTROM-Meter	Mandatory

Response

HTTP Status 200

```
result array of digitalSTROM Meters
```

Sample

```

GET /json/apartment/removeMeter?dsuid=3504175fe0000000000000000000000016c4f00
{
  "ok": true
}

```

2.6 Sensors

2.6.1 Get Assigned Sensors

Returns the list of assigned sensor devices in all zones.

Synopsis

HTTP GET /json/apartment/getAssignedSensors

Parameter

None

Response

HTTP Status 200

id	Id of this zone
name	Name of this zone
sensorType	Numerical value of the sensor type
dsuid	dSUID of the source device

Sample

```
GET /json/apartment/getAssignedSensors
{
  "ok":true,
  "result":
    "zones": [
      {
        "id": 1,
        "name": "LivingRoom",
        "sensors": [
          {
            "sensorType": 9,
            "dsuid": 3504175fe000000000000000000000016be700
          },
          {
            "sensorType": 11,
            "dsuid": 3504175fe000000000000000000000016be700
          }
        ]
      },
      {
        "id": 2,
        "name": "Kitchen",
        "sensors": [
          {
            "sensorType": 9,
            "dsuid": 3504175fe00000000000000000000001456700
          }
        ]
      }
    ]
}
```

2.6.2 Get Sensor Values

Returns a list of sensor relevant for the apartment.

For the apartment the temperature, humidity, and brightness are sensor types that are tracked. Additionally there is

For each zone the temperature, humidity, CO2 concentration and brightness are sensor types that are tracked. Typically there is one device as a zone reference for these values.

If there is no standard device defined for a sensor type or if no measurement is available there is neither the value or time field returned.

Synopsis

HTTP GET /json/apartment/getSensorValues

Parameter

None

Response

HTTP Status 200

The result object contains the following outdoor measurements:

temperature	Temperature value and timestamp of last measurement
humidity	Humidity value and timestamp of last measurement
brightness	Brightness value and timestamp of last measurement
precipitation	Precipitation value and timestamp of last measurement
airpressure	Air pressure value and timestamp of last measurement
windspeed	Wind speed value and timestamp of last measurement
winddirection	Wind direction in degrees value and timestamp of last measurement
gustspeed	Gust speed value and timestamp of last measurement
gustdirection	Gust direction in degrees value and timestamp of last measurement

If there is external weather service data available from my.digitalSTROM for the geo location of the installation it will be provided as well:

WeatherIconId
WeatherConditionId
WeatherServiceId
WeatherServiceTime

The result object contains a "zones" field with an array of all zones of the apartment and the relevant sensor data:

TemperatureValue	Temperature value
TemperatureValueTime	Timestamp of the temperature measurement
HumidityValue	Humidity value
HumidityValueTime	Timestamp of the humidity measurement
CO2ConcentrationValue	CO2Concentration value
CO2ConcentrationValueTime	Timestamp of the CO2 concentration measurement
BrightnessValue	Brightness value
BrightnessValueTime	Timestamp of the brightness measurement

Sample

```
GET /json/apartment/getSensorValues
{
  "ok": true,
  "result": {
    "weather": {
      "WeatherIconId": "04d",
      "WeatherConditionId": "803",
      "WeatherServiceId": "7",
      "WeatherServiceTime": "2017-03-20T14:33:50.328Z"
    },
    "outdoor": {
      "temperature": {
        "value": 20.975,
        "time": "2017-03-20T13:53:15.603Z"
      }
    }
  }
}
```

```

    },
    "humidity": {
      "value": 71,
      "time": "2017-03-20T13:53:15.603Z"
    },
    "windspeed": {
      "value": 1,
      "time": "2017-03-20T14:33:29.946Z"
    },
    "winddirection": {
      "value": 0,
      "time": "2017-03-20T14:33:29.943Z"
    },
    "gustspeed": {
      "value": 7.2,
      "time": "2017-03-20T14:33:29.947Z"
    },
    "gustdirection": {
      "value": 0.25,
      "time": "2017-03-20T14:33:29.947Z"
    },
    "precipitation": {
      "value": 0,
      "time": "2017-03-20T14:33:29.948Z"
    },
    "airpressure": {
      "value": 1010,
      "time": "2017-03-20T14:33:29.761Z"
    }
  },
  "zones": [
    {
      "id": 1142,
      "name": "Küche",
      "values": []
    },
    {
      "id": 1168,
      "name": "Wohnzimmer",
      "values": [
        {
          "TemperatureValue": 22.55,
          "TemperatureValueTime": "2014-10-13T18:07:24.528+0200"
        },
        {
          "HumidityValue": 59.2,
          "HumidityValueTime": "2014-10-13T18:07:24.638+0200"
        },
        {
          "CO2ConcentrationValue": 1209.205182943208,
          "CO2ConcentrationValueTime": "2014-10-13T11:45:53.756+0200"
        }
      ]
    },
    {
      "id": 1191,
      "name": "Galerie",
      "values": [
        {
          "TemperatureValue": 21.75,
          "TemperatureValueTime": "2014-10-13T18:04:13.382+0200"
        },
        {
          "HumidityValue": 64.4,
          "HumidityValueTime": "2014-10-13T18:04:13.480+0200"
        }
      ]
    },
    {
      "id": 1192,
      "name": "Flur",
      "values": [
        {
          "TemperatureValue": 21.950000000000005,
          "TemperatureValueTime": "2014-10-13T18:04:10.022+0200"
        }
      ]
    }
  ]
},

```

```

    {
      "id": 3,
      "name": "Wintergarten",
      "values": [
        {
          "TemperatureValue": 19.67500000000001,
          "TemperatureValueTime": "2014-10-13T18:06:07.659+0200"
        },
        {
          "HumidityValue": 66,
          "HumidityValueTime": "2014-10-13T18:06:07.790+0200"
        }
      ]
    },
    {
      "id": 10,
      "name": "Terrasse",
      "values": []
    }
  ]
}

```

2.6.3 Set Weather Information

Synopsis

HTTP GET /json/apartment/setWeatherInformation

Parameter

Parameter	Description	Remarks
icon	string id, last character d / n indicates day / night	Mandatory
condition	string id, see https://openweathermap.org/weather-conditions	Mandatory
service	Service id	Optional
ts	string timestamp	Optional

Sample

```

GET /json/apartment/setWeatherInformation?icon=10n&condition=500'
{
  "ok":true
}

```

2.7 Heating

2.7.1 Get Temperature Control Status

Get the current status of temperature control in all zones.

Synopsis

HTTP GET /json/apartment/getTemperatureControlStatus

Parameter

None

Response

HTTP Status 200

id	Id of the zone
name	Name of the zone
ControlMode	Control mode: 0=off; 1=pid-control; 2=zone-follower; 3=fixed-value; 4=manual
OperationMode	Current operation mode of the control
TemperatureValue	Current temperature of the zone
TemperatureValueTime	Timestamp of last temperature data update, seconds since epoch
NominalValue	Target temperature of this zone
NominalValueTime	Timestamp of last set point change, seconds since epoch
ControlValue	Current control value
ControlValueTime	Timestamp of last control value data update, seconds since epoch

Sample

```
GET /json/apartment/getTemperatureControlStatus
{
  "ok": true,
  "result": {
    "zones": [
      {
        "id": 1,
        "name": "LivingRoom",
        "ControlMode": 1,
        "OperationMode": 4,
        "TemperatureValue": 20.7,
        "NominalValue": 20.0,
        "ControlValue": 92.5,
        "TemperatureValueTime": 2014-10-08T18:21:05Z,
        "NominalValueTime": 2014-10-08T18:00:00Z,
        "ControlValueTime": 2014-10-08T18:22:00Z
      },
      {
        "id": 2,
        "name": "Kitchen",
        "ControlMode": 2,
        "ControlValue": 92.5,
        "ControlValueTime": 2014-10-08T18:19:00Z
      },
      {
        "id": 3,
        "name": "Corridor",
        "ControlMode": 3,
        "OperationMode": 2,
        "ControlValue": 80
      }
    ]
  }
}
```

2.7.2 Get Temperature Control Configuration

Get the configuration of the temperature control settings for all zones.

Synopsis

HTTP GET /json/apartment/getTemperatureControlConfig

Parameter

None

Response

HTTP Status 200

id	Id of this zone
name	Name of this zone
ControlMode	Control mode: 0=off; 1=pid-control; 2=zone-follower; 3=fixed-offset; 4=manual
ManualValue	Fixed control value for manual mode (mode 4 only)
ReferenceZone	Zone number of the reference zone (mode 2 only)
CtrlOffset	Static control value offset (mode 2 only)
EmergencyValue	Fixed control value in case of malfunction (mode 1 only)
CtrlKp	Control proportional factor
CtrlTs	Control sampling time
CtrlTi	Control integrator time constant
CtrlKd	Control differential factor
CtrlImin	Control minimum integrator value
CtrlImax	Control maximum integrator value
CtrlYmin	Control minimum control value
CtrlYmax	Control maximum control value
CtrlAntiWindUp	Control integrator anti wind up: 0=inactive, 1=active

Sample

```
GET /json/apartment/getTemperatureControlConfig
```

```
{
  "ok": true,
  "result": {
    "zones": [
      {
        "id": 1,
        "name": "LivingRoom",
        "ControlMode": 1,
        "EmergencyValue": 50,
        "CtrlKp": 5.2,
        "CtrlTs": 240,
        "CtrlTi": 1,
        "CtrlKd": 1,
        "CtrlImin": 600,
        "CtrlImax": 2400,
        "CtrlYmin": 0,
        "CtrlYmax": 100,
        "CtrlAntiWindUp": 1
      },
      {
        "id": 2,
        "name": "Kitchen",
        "ControlMode": 2,
        "ReferenceZone": 0,
        "CtrlOffset": -10
      },
      ...
      {
        "id": 3,
        "name": "Corridor",
        "ControlMode": 3
      }
    ]
  }
}
```



```

    }
  ]
}

```

2.7.3 Get Temperature Control Values

Returns a list of all temperature control preset values of all zones. Every control operation mode has up to 15 presets defined, where 6 of them are actually used by the system.

Synopsis

HTTP GET /json/apartment/getTemperatureControlValues

Parameter

None

Response

HTTP Status 200

id	Id of this zone
name	Name of this zone
Off	Preset value for operation mode 0: "Off"
Comfort	Preset value for operation mode 1: "Comfort"
Economy	Preset value for operation mode 2: "Economy"
NotUsed	Preset value for operation mode 3: "Not Used"
Night	Preset value for operation mode 4: "Night"
Holiday	Preset value for operation mode 5: "Holiday"
Cooling	Preset value for operation mode 6: "Cooling"
CoolingOff	Preset value for operation mode 7: "CoolingOff"

Sample

GET /json/apartment/getTemperatureControlValues

```

{
  "ok": true,
  "result": {
    "zones": [
      {
        "id": 1,
        "name": "LivingRoom",
        "Off": 6,
        "Comfort": 21,
        "Economy": 20,
        "NotUsed": 18,
        "Night": 16,
        "Holiday": 12,
        "Cooling": 23,
        "CoolingOff": 50,
      },
      ...
      {
        "id": 972,
        "name": "",
        "Off": 8,
        "Comfort": 22,
        "Economy": 20,
        "NotUsed": 18,
        "Night": 17,
        "Holiday": 16,
      }
    ]
  }
}

```

```

    "Cooling": 23,
    "CoolingOff": 50
  }
}

```

2.7.4 Get Temperature Control Configuration v2

Get the temperature control configuration parameters for each zone with one call.

Synopsis

HTTP GET /json/apartment/getTemperatureControlConfig2

Parameter

None

Response

HTTP Status 200

id	Id of this zone
name	Name of this zone
mode	Current Control Mode of the zone: "off", "control", "zoneFollower", "fixed", "manual"
targetTemperatures	Set point temperatures for each operation mode of the zone
fixedValues	Fixed control values for each operation mode of the zone
controlMode	Object with the PID controller related parameters
zoneFollowerMode	Object with the zone follower related parameters
manualMode	Object with the manual mode parameter control value

controlMode

emergencyValue	Fixed control value in case of malfunction
ctrlKp	Control proportional factor
ctrlTs	Control sampling time
ctrlTi	Control integrator time constant
ctrlKd	Control differential factor
ctrlImin	Control minimum integrator value
ctrlImax	Control maximum integrator value
ctrlYmin	Control minimum control value
ctrlYmax	Control maximum control value
ctrlAntiWindUp	Control integrator anti wind up

zoneFollowerMode

referenceZone	Zone number of the reference zone
ctrlOffset	Control value offset

manualMode

controlValue	Control value for manual mode
--------------	-------------------------------

Sample

```
GET /json/apartment/getTemperatureControlConfig?id=1237
{
  "ok": true,
  "result": {
    "zones": {
      "1": {
        "name": "LivingRoom",
        "config": {
          "mode": "manual",
          "targetTemperatures": {
            "0": 6, "1": 23.5, "2": 22, "3": 19,
            "4": 18, "5": 18, "6": 22, "7": 50,
            "8": 24, "9": 28, "10": 32, "11": 30
          },
          "fixedValues": {
            "0": 0, "1": 100, "2": 90, "3": 80,
            "4": 70, "5": 25, "6": 100, "7": 0,
            "8": 80, "9": 60, "10": 40, "11": 25
          },
          "controlMode": {
            "emergencyValue": 50,
            "ctrlKp": 5.2,
            "ctrlTs": 240,
            "ctrlTi": 1,
            "ctrlKd": 1,
            "ctrlImin": 600,
            "ctrlImax": 2400,
            "ctrlYmin": 0,
            "ctrlYmax": 100,
            "ctrlAntiWindUp": 1
          },
          "zoneFollowerMode": {
            "referenceZone": 38523,
            "ctrlOffset": 10
          },
          "manualMode": {
            "controlValue": 30
          }
        }
      }
    }
  }
}
```

2.8 DevicesFirstSeen

2.8.1 setDevicesFirstSeen

Sets the FirstSeen property of all devices with first seen date before 1 January 2011. All other devices are not modified

Synopsis

HTTP GET /json/apartment/setDevicesFirstSeen

Parameter	Description	Remarks
time	ISO8601 time when the devices were registered	Mandatory

Parameter

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/apartment/setDevicesFirstSeen?time=2012-06-14T10:42:13Z
{
  "ok":true
}
```

2.9 ModelFeatures

ModelFeatures are used to determine the visibility and (to some extent) the functionality of the Configurator-UI.

2.9.1 getModelFeatures

Returns the known ModelFeatures.

Synopsis

HTTP GET /json/apartment/getModelFeatures

Parameter	Description	Remarks
-----------	-------------	---------

Parameter

Response

HTTP Status 200

result.modelFeatures	object containing the known model features. The features are ordered according to the device's color (e.g. "GE", "SW", etc.). Always the most specific model feature applies: e.g. (refer to the example below) "KM:200" from "GE" applies to a GE-KM220 device; while a "KM:2" from "GE" applies to a GE-KM210 device.
result.modelFeatures.<color>	
result.modelFeatures.<color>.<model>	
reference	object containing all defined model features

Sample

```
GET /json/apartment/getModelFeatures
{
  "ok": true,
  "result": {
    "modelFeatures": {
      "GE": {
        "KM:220": {
          "dontcare": true,
          "blink": true,
          "ledauto": true
        },
        "KM:2": {
          "dontcare": true,
```

```

        "blink": true,
        "ledauto": true
    },
    "KL:200": {
        "dontcare": true,
        "blink": true
    }
},
"GR": {
    "KL:210": {
        "dontcare": true,
        "blink": true,
        "ledauto": true
    }
    "KL:2": {
        "dontcare": true,
        "blink": true
    }
}
},
"reference": {
    "dontcare": false,
    "blink": false,
    "ledauto": false,
    "leddark": false
}
}
}

```

2.10 MuteFire

Mutes the fire alarm. Does nothing if fire alarm is not active or if fire mute is not enabled in digitalSTROM-Server.

See also fire, fireMuteEnabled states in the system-interfaces document.

Synopsis

HTTP GET /json/apartment/muteFire

Response

HTTP Status 200

ok	true
----	------

Sample

```

GET /json/apartment/muteFire
{
  "ok":true
}

```

2.11 Controller Statuses

2.11.1 getControllerStatus

Polls status of all controllers in the system.

Synopsis

HTTP GET /json/apartment/getControllerStatuses

Parameter

Parameter	Description	Remarks
lang	Locale Code based on Language_Region pattern, e.g. en_EN, de_DE	

Response

HTTP Status 200

code	off ready error
message	Translated message for user providing further details

Sample

```
GET /json/apartment/getControllerStatuses
{
  "ok":true,
  "result":{
    "controllerStatuses":{
      "302ed89f43f0000000000e400000605700":{"code":"ready","message":"Ready"},
      "3504175fe0000000000000100000071700":{"code":"ready","message":"Ready"},
      "7f40dee794044465a38bcd7457094e300":{"code":"ready","message":"Authorization will expire in 1 week."}
    }
  }
}
```

3 Zone

3.1 Common

Every `/json/zone/` function uses a common selection scheme for the zone to which the command refers to. Either the parameter "id" or "name" must be given to identify the zone. The special value zero for the "id" maybe used to send the command as broadcast to all zones.

Parameter	Description	Remarks
id	Zone Number	Optional
name	Zone Name	Optional

A missing zone identifier result in the following error message to be returned.

```
{
  "ok": false,
  "message": "Need parameter name or id to identify zone"
}
```

If a zone identifier does not match any actually known zone in the installation the following error message is returned.

```
{
  "ok": false,
  "message": "Could not find zone with id '1250'"
}
```

3.2 Action

3.2.1 getActions

Get the actions for specified application.

Synopsis

HTTP GET `/json/zone/getActions`

Parameter

Parameter	Description	Remarks
application	Number of the target application	Mandatory
lang	Locale Code based on Language_Region pattern, e.g. en_EN, de_DE	Optional

Response

HTTP Status 200

```
result.actions Array of actions with action.id and action.title
```

If the lang parameter is omitted the action.title is translated to english as default.

Sample

```
GET /json/zone/getActions?id=0&application=2
```

```
{
  "result":{
    "actions":[
      {
        "id":"app.moveUp",
        "title":"Blinds[]Move[]Up"
      },
      {
        "id":"app.moveDown",
        "title":"Blinds[]Move[]Down"
      },
      {
        "id":"app.stepUp",
        "title":"Blinds[]Step[]Up"
      },
      {
        "id":"app.stepDown",
        "title":"Blinds[]Step[]Down"
      },
      {
        "id":"app.sunProtection",
        "title":"Blinds[]Sun[]Protection"
      },
      {
        "id":"app.stop",
        "title":"Blinds[]Stop"
      }
    ]
  },
  "ok":true
}
```

Sample

Example with lang parameter

```
GET /json/zone/getActions?id=0&application=2&lang=de_DE
```

```
{
  "result":{
    "actions":[
      {
        "id":"app.moveUp",
        "title":"Rollläden[]Hochfahren"
      },
      {
        "id":"app.moveDown",
        "title":"Rollläden[]Herunterfahren"
      },
      {
        "id":"app.stepUp",
        "title":"Rollläden[]1[]Schritt[]Hochfahren"
      },
      {
        "id":"app.stepDown",
        "title":"Rollläden[]1[]Schritt[]Runterfahren"
      },
      {
        "id":"app.sunProtection",
        "title":"Sonnenschutz[]Für[]Rollläden"
      },
      {
        "id":"app.stop",
        "title":"Rollläden[]Stoppen"
      }
    ]
  },
  "ok":true
}
```


3.2.2 callAction

Execute the action in a zone or in a cluster for devices of specified application.

Synopsis

HTTP GET /json/zone/callAction

Parameter

Parameter	Description	Remarks
action	Name of the action to be executed	Mandatory
application	Number of the target application	Conditional
cluster	Name of the target cluster	Optional

The application parameter can be omitted only if cluster exists. To call action within apartment, param id must be equal to 0.

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/zone/callAction?id=0&action=app.moveUp&application=2
{
  "ok":true
}
```

Example with the cluster parameter.

```
GET /json/zone/callAction?id=0&action=app.moveUp&cluster=17
{
  "ok":true
}
```

List of all supported actions

Id	Name	Application Name	Application
app.moveUp	Blinds Move Up	Blinds	2
app.moveDown	Blinds Move Down	Blinds	2
app.stepUp	Blinds Step Up	Blinds	2
app.stepDown	Blinds Step Down	Blinds	2
app.sunProtection	Blinds Sun Protection	Blinds	2
app.stop	Blinds Stop	Blinds	2
app.moveIn	Awnings Move In	Awnings	65
app.moveOut	Awnings Move Out	Awnings	65
app.stepIn	Awnings Step In	Awnings	65
app.stepOut	Awnings Step Out	Awnings	65

3.3 Name

3.3.1 getName

Returns the user defined name of the zone.

Synopsis

HTTP GET /json/zone/getName

Parameter

None

Response

HTTP Status 200

name	identifier string for the zone
------	--------------------------------

Sample

```
GET /json/zone/getName?id=1237
{
  "ok":true,
  "result" :
  {
    "name" : "Wohnen"
  }
}
```

3.3.2 setName

Sets the zone name.

Synopsis

HTTP GET /json/zone/setName

Parameter	Description	Remarks
newName	identifier string for the zone	Mandatory

Parameter

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/zone/setName?id=1237&newName="Wohnen"
{
  "ok":true
}
```

3.4 Scene

3.4.1 callScene

Executes the scene sceneNumber in a zone for a group of devices.

Synopsis

HTTP GET /json/zone/callScene

Parameter

Parameter	Description	Remarks
sceneNumber	Numerical value	Mandatory
groupID	Number of the target group	Optional
groupName	Name of the target group	Optional
force	Boolean value, if set a forced scene call is issued	Optional

If the group parameters are omitted the command is sent as broadcast to all devices in a zone.

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/zone/callScene?id=1237&groupID=1&sceneNumber=5&force=true
{
  "ok":true
}
```

3.4.2 saveScene

Tells devices to store their current output values as a default for the scene sceneNumber.

Synopsis

HTTP GET /json/zone/saveScene

Parameter

Parameter	Description	Remarks
sceneNumber	Numerical value	Mandatory
groupID	Number of the target group	Optional
groupName	Name of the target group	Optional

If the group parameters are omitted the command is sent as broadcast to all devices in a zone.

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/zone/saveScene?id=1237&groupID=2&sceneNumber=17
{
  "ok":true
}
```

3.4.3 undoScene

Tells devices to restore their output values to the previous state if the current scene matches the sceneNumber.

Synopsis

HTTP GET /json/zone/undoScene

Parameter

Parameter	Description	Remarks
sceneNumber	Numerical value	Mandatory
groupID	Number of the target group	Optional
groupName	Name of the target group	Optional

If the group parameters are omitted the command is sent as broadcast to all devices in the zone.

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/zone/undoScene?id=1237&sceneNumber=65
{
  "ok":true
}
```

3.4.4 sceneGetName

Get the user defined name for a scene sceneNumber within a group of a zone.

Synopsis

HTTP GET /json/zone/sceneGetName

Parameter

Parameter	Description	Remarks
sceneNumber	Numerical value	Mandatory
groupID	Number of the target group	M/O
groupName	Name of the target group	M/O

Either groupID or groupName must be supplied to this request.

Response

HTTP Status 200

```
result.name the user defined name of the scene
```

Sample

```
GET /json/zone/sceneGetName?id=1237&sceneNumber=19&groupID=1
{
  "ok":true
  result:{
    "name":"Fernsehen"
  }
}
```

3.4.5 sceneSetName

Sets a user defined name for a scene sceneNumber within a group of a zone. This name is stored on the digitalSTROM-Server only.

Synopsis

HTTP GET /json/zone/sceneSetName

Parameter

Parameter	Description	Remarks
newName	User defined name of the scene	Mandatory
sceneNumber	Numerical value	Mandatory
groupID	Number of the target group	M/O
groupName	Name of the target group	M/O

Either groupID or groupName must be supplied to this request.

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/zone/sceneSetName?id=1237&sceneNumber=17&groupID=2&newName="Fernsehen"
{
  "ok":true
}
```

3.4.6 getReachableScenes

Returns a list of groups which can be controlled by pushbuttons which are actually present in the zone.

Synopsis

HTTP GET /json/zone/getReachableScenes

Parameter

Parameter	Description	Remarks
groupID	Number of the target group	Mandatory
groupName	Name of the target group	Optional

Either groupID or groupName are required.

Response

HTTP Status 200

```
result.reachableScens  array of scene numbers
result.userSceneNames  array of user defined scene names
```

Sample

```
GET /json/zone/getReachableScenes?id=1237&groupID=1
{
  "ok": true,
  "result": {
    "reachableScenes": [
      0,
      1,
      5,
      6,
      17,
      18,
      19,
      29,
      30,
      31,
      38,
      39
    ]
    "userSceneNames": [
      {
        "sceneNr": 17,
        "sceneName": "Watch TV"
      },
      {
        "sceneNr": 18,
        "sceneName": "Reading"
      }
    ]
  }
}
```

3.4.7 getLastCalledScene

Returns the sceneNumber which has been executed last for a group in a zone.

Synopsis

HTTP GET /json/zone/getLastCalledScene

Parameter

Parameter	Description	Remarks
groupID	Number of the target group	Optional
groupName	Name of the target group	Optional

Response

HTTP Status 200

```
result.scene the number of the last called scene
```

Sample

```
GET /json/zone/getLastCalledScene?id=1237&groupID=1
{
  "ok": true,
  "result": {
    "scene": 17
  }
}
GET /json/zone/getLastCalledScene?id=0
{
  "ok": true,
  "result": {
    "scene": 69
  }
}
```

3.5 Value

3.5.1 Set Output Value

Set the output value of a group of devices in a zone to a given value.

Notice Setting output values directly bypasses the group state machine and is not recommended.

Synopsis

HTTP GET /json/zone/setValue

Parameter

Parameter	Description	Remarks
value	Numerical value	Mandatory
groupID	Number of the target group	Optional
groupName	Name of the target group	Optional

If the group parameters are omitted the command is sent as broadcast to all devices in the selected zone.

Notice Setting output values without a group identification is strongly unrecommended.

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/zone/setValue?id=1237&value=0&groupID=2
{
  "ok":true,
}
```

3.5.2 Blink

Executes the "blink" function on a group of devices in a zone for identification purposes.

Synopsis

HTTP GET /json/zone/blink

Parameter

Parameter	Description	Remarks
groupID	Number of the target group	Optional
groupName	Name of the target group	Optional

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/zone/blink?id=1237&groupID=1
{
  "ok":true,
}
```

3.5.3 Send Sensor Value

Send a sensor value to a group of devices in a zone.

Synopsis

HTTP GET /json/zone/pushSensorValue

Parameter

Parameter	Description	Remarks
groupID	Number of the target group	Default "0", optional
sourceDSUID	DSUID of the originating device	Optional
sensorValue	Numerical value	Mandatory
sensorType	Numerical type of the sensor	Mandatory

If the group parameter is omitted the command is sent as broadcast to all devices in the selected zone. The reference for the sensor type definitions can be found in the ds-basics document.

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/zone/pushSensorValue?id=1237&sensorType=51&sensorValue=100&groupID=48
{
  "ok":true,
}
```

3.5.4 Set Status Field

Set the value of a group application status.

Synopsis

HTTP GET /json/zone/setStatusField

The following fields and attributes are defined:

Attribute	Application	Remarks
malfunction	Apartment Ventilation (groupID 64)	Indicates malfunction of the whole service or a device
service	Apartment Ventilation (groupID 64)	Indicates service request for a device

The group status flags are available as status object in the property tree:

```
/json/property/query?query=/usr/states/zone.0.group.64.status.service(*)
```

The malfunction and service attributes can also be set by hardware sensor inputs.

Parameter

Parameter	Description	Remarks
groupID	Number of the target group	Default "0", optional
field	Field name	Mandatory
value	String value of the attribute	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/zone/setStatusField?id=0&groupID=64&field=malfunction&value=active
{
  "ok":true,
}
```

3.6 Sensors

3.6.1 Set Sensor Source

Set the source of a sensor in a zone to a given device source address. For example one might have multiple temperature and humidity sensors in a room and using this method he can select which one to use for room temperature control.

Synopsis

HTTP GET /json/zone/setSensorSource

Parameter

Parameter	Description	Remarks
dsuid	dSUID of the source device	Mandatory
sensorType	Numerical value of the sensor type	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/zone/setSensorSource?id=1237&sensorType=9&dsuid=3504175fe0000000000000000000000016be700
{
  "ok": true,
}
```

3.6.2 Clear Sensor Source

Remove all assignments for a particular sensor type in a zone.

Synopsis

HTTP GET /json/zone/clearSensorSource

Parameter

Parameter	Description	Remarks
sensorType	Numerical value of the sensor type	Mandatory

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/zone/clearSensorSource?id=1237&sensorType=11
{
  "ok": true,
}
```

3.6.3 Get Assigned Sensors

Returns the list of assigned sensor devices in a zone.

Synopsis

HTTP GET /json/zone/getAssignedSensors

Parameter

None

Response

HTTP Status 200

sensorType	Numerical value of the sensor type
dsuid	dSUID of the source device

Sample

```
GET /json/zone/getAssignedSensors?id=1237
{
  "ok": true,
  "result": {
    "sensors": [
      {
        "sensorType": 9,
        "dsuid": 3504175fe000000000000000000000016be700
      },
      {
        "sensorType": 11,
        "dsuid": 3504175fe000000000000000000000016be700
      }
    ]
  }
}
```

3.6.4 Get Sensor Values

Returns a list of sensor measurements relevant for a zone. The temperature, humidity, CO2 concentration and brightness are sensor types that are tracked for a zone. Typically there is one device as a zone reference for these values.

If there is no standard device defined for a sensor type or if no measurement is available there is neither the value or time field returned.

Synopsis

HTTP GET /json/zone/getSensorValues

Parameter

None

Response

HTTP Status 200

TemperatureValue	Temperature value
TemperatureValueTime	Timestamp of the temperature measurement
HumidityValue	Humidity value
HumidityValueTime	Timestamp of the humidity measurement
CO2ConcentrationValue	CO2Concentration value
CO2ConcentrationValueTime	Timestamp of the CO2 concentration measurement
BrightnessValue	Brightness value
BrightnessValueTime	Timestamp of the brightness measurement

Sample

```
GET /json/zone/getSensorValues?id=1237
{
  "ok": true,
  "result": {
    "id": 14236,
    "name": "Heizungsraum",
    "values": [
      {
        "TemperatureValue": 26.5,
        "TemperatureValueTime": "2014-10-13T17:57:21.445+0200"
      }
    ]
  }
}
```

3.7 Heating

3.7.1 Get Temperature Control Status

Get the current status of the zone temperature control.

Synopsis

HTTP GET /json/zone/getTemperatureControlStatus

Parameter

None

Response

HTTP Status 200

ControlMode	Control mode: 0=off; 1=pid-control; 2=zone-follower; 3=fixed-value; 4=manual
OperationMode	Current operation mode of the control
TemperatureValue	Current temperature of the zone
TemperatureValueTime	Timestamp of last temperature data update, seconds since epoch
NominalValue	Target temperature of this zone
NominalValueTime	Timestamp of last set point change, seconds since epoch
ControlValue	Current control value
ControlValueTime	Timestamp of last control value data update, seconds since epoch

Sample

```
GET /json/zone/getTemperatureControlStatus?id=1237
{
  "ok": true,
  "result":
  {
    "ControlMode": 1,
    "OperationMode": 4,
    "TemperatureValue": 20.7,
    "NominalValue": 20.0,
    "ControlValue": 92.5,
    "TemperatureValueTime": 2014-10-08T18:21:05Z,
    "NominalValueTime": 2014-10-08T18:00:00Z,
    "ControlValueTime": 2014-10-08T18:22:00Z
  }
}
```

3.7.2 Get Temperature Control Configuration

Get the configuration of the zone temperature control.

Synopsis

HTTP GET /json/zone/getTemperatureControlConfig

Parameter

None

Response

HTTP Status 200

ControlMode	Control mode: 0=off; 1=pid-control; 2=zone-follower; 3=fixed-value; 4=manual
ReferenceZone	Zone number of the reference zone (mode 2 only), can be zero
CtrlOffset	Control value offset (mode 2 only)
ManualValue	Fixed control value for manual mode (mode 4 only)
EmergencyValue	Fixed control value in case of malfunction (mode 1 only)
CtrlKp	Control proportional factor
CtrlTs	Control sampling time
CtrlTi	Control integrator time constant
CtrlKd	Control differential factor
CtrlImin	Control minimum integrator value
CtrlImax	Control maximum integrator value
CtrlYmin	Control minimum control value
CtrlYmax	Control maximum control value
CtrlAntiWindUp	Control integrator anti wind up: 0=inactive, 1=active

Sample

```
GET /json/zone/getTemperatureControlConfig?id=1237
{
  "ok": true,
  "result":
  {
    "ControlMode": 1,
    "EmergencyValue": 50,
    "CtrlKp": 5.2,
    "CtrlTs": 240,
    "CtrlTi": 1,
    "CtrlKd": 1,
    "CtrlImin": 600,
    "CtrlImax": 2400,
    "CtrlYmin": 0,
    "CtrlYmax": 100,
    "CtrlAntiWindUp": 1
  }
}
```

```
GET /json/zone/getTemperatureControlConfig?id=1237
{
  "ok": true,
  "result":
  {
    "ControlMode": 2,
    "ReferenceZone": 0,
    "CtrlOffset": 10
  }
}
```

```
GET /json/zone/getTemperatureControlConfig?id=1237
{
  "ok": true,
  "result":
  {
    "ControlMode": 3
  }
}
```

```

GET /json/zone/getTemperatureControlConfig?id=1237
{
  "ok": true,
  "result":
  {
    "ManualValue": 87.5,
    "ControlMode": 4
  }
}

```

3.7.3 Set Temperature Control Configuration

Set the configuration of the zone temperature control.

Synopsis

HTTP GET /json/zone/setTemperatureControlConfig

Parameter

ControlMode	Control mode, can be one of: 0=off; 1=pid-control; 2=zone-follower; 3=fixed-value; 4=manual	Optional
ReferenceZone	Zone number of the reference zone	Optional for ControlMode 2
CtrlOffset	Control value offset	Optional for ControlMode 2
EmergencyValue	Fixed control value in case of malfunction	Optional for ControlMode 1
ManualValue	Control value for manual mode	Optional for ControlMode 4
CtrlKp	Control proportional factor	Mandatory for ControlMode 1
CtrlTs	Control sampling time	Mandatory for ControlMode 1
CtrlTi	Control integrator time constant	Mandatory for ControlMode 1
CtrlKd	Control differential factor	Mandatory for ControlMode 1
CtrlImin	Control minimum integrator value	Mandatory for ControlMode 1
CtrlImax	Control maximum integrator value	Mandatory for ControlMode 1
CtrlYmin	Control minimum control value	Optional for ControlMode 1
CtrlYmax	Control maximum control value	Optional for ControlMode 1
CtrlAntiWindUp	Control integrator anti wind up	Mandatory for ControlMode 1

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/zone/setTemperatureControlConfig?id=1237&ControlMode=2&ReferenceZone=4327&CtrlOffset=-20
{
  "ok": true,
}
```

3.7.4 Get Temperature Control Values

Get the temperature control operation mode preset values for a zone. Every control operation mode has up to 15 presets defined.

Synopsis

HTTP GET /json/zone/getTemperatureControlValues

Parameter

None

Response

HTTP Status 200

Off	Preset value for operation mode 0: "Off"
Comfort	Preset value for operation mode 1: "Comfort"
Economy	Preset value for operation mode 2: "Economy"
NotUsed	Preset value for operation mode 3: "Not Used"
Night	Preset value for operation mode 4: "Night"
Holiday	Preset value for operation mode 5: "Holiday"
Cooling	Preset value for operation mode 6: "Cooling"
CoolingOff	Preset value for operation mode 7: "CoolingOff"

Sample

```
GET /json/zone/getTemperatureControlValues?id=1237
{
  "ok": true,
  "result":
  {
    "Off": 22.5,
    "Comfort": 21,
    "Economy": 20,
    "NotUsed": 18,
    "Night": 16,
    "Holiday": 4,
    "Cooling": 23,
    "CoolingOff": 50,
  }
}
```


3.7.5 Set Temperature Control Values

Set the temperature control operation mode preset values for a zone. Single values can be given and others that do not change may be omitted.

Notice For operation mode "PID Control" the given values are nominal temperatures, and for operation mode "Fixed Values" the given values are absolute control values.

Synopsis

HTTP GET /json/zone/setTemperatureControlValues

Parameter

Off	Preset value for operation mode 0: "Off"	Optional
Comfort	Preset value for operation mode 1: "Comfort"	Optional
Economy	Preset value for operation mode 2: "Economy"	Optional
NotUsed	Preset value for operation mode 3: "Not Used"	Optional
Night	Preset value for operation mode 4: "Night"	Optional
Holiday	Preset value for operation mode 5: "Holiday"	Optional
Cooling	Preset value for operation mode 6: "Cooling"	Optional
CoolingOff	Preset value for operation mode 7: "CoolingOff"	Optional

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/zone/setTemperatureControlValues?id=1237&Comfort=22.5&Night=21
{
  "ok": true
}
```

3.7.6 Get Temperature Control Configuration v2

Get the temperature control configuration parameters.

Synopsis

HTTP GET /json/zone/getTemperatureControlConfig2

Parameter

None

Response

HTTP Status 200

mode	Current Control Mode of the zone: off", "control", "zoneFollower", "fixed", "manual"
targetTemperatures	Set point temperatures for each operation mode of the zone
fixedValues	Fixed control values for each operation mode of the zone
controlMode	Object with the PID controller related parameters
zoneFollowerMode	Object with the zone follower related parameters
manualMode	Object with the manual mode parameter control value

controlMode

emergencyValue	Fixed control value in case of malfunction (int)
ctrlKp	Regulation proportional factor (double)
ctrlTs	Regulation sampling time (int)
ctrlTi	Regulation integrator time constant (int)
ctrlKd	Regulation differential factor (int)
ctrlImin	Regulation minimum integrator value (double)
ctrlImax	Regulation maximum integrator value (double)
ctrlYmin	Regulation minimum control value (int)
ctrlYmax	Regulation maximum control value (int)
ctrlAntiWindUp	Regulation integrator anti wind up (bool)

zoneFollowerMode

referenceZone	Zone number of the reference zone (int)
ctrlOffset	Control value offset (int)

manualMode

controlValue	Control value for manual mode (int)
--------------	-------------------------------------

Sample

```
GET /json/zone/getTemperatureControlConfig?id=1237
{
  "mode": "control",
  "targetTemperatures": {
    "0": 6, "1": 23.5, "2": 22, "3": 19,
    "4": 18, "5": 18, "6": 22, "7": 50,
    "8": 24, "9": 28, "10": 32, "11": 30
  },
  "fixedValues": {
    "0": 0, "1": 100, "2": 90, "3": 80,
    "4": 70, "5": 25, "6": 100, "7": 0,
    "8": 80, "9": 60, "10": 40, "11": 25
  },
  "controlMode": {
    "emergencyValue": 50,
    "ctrlKp": 5.2,
    "ctrlTs": 240,
    "ctrlTi": 1,
    "ctrlKd": 1,
    "ctrlImin": 600,
    "ctrlImax": 2400,
    "ctrlYmin": 0,
    "ctrlYmax": 100,
  }
}
```

```

    "ctrlAntiWindUp": true
  },
  "zoneFollowerMode" : {
    "referenceZone": 38523,
    "ctrlOffset": 10
  },
  "manualMode" : {
    "controlValue" : 30
  }
}

```

3.7.7 Set Temperature Control Configuration v2

Set the configuration of the zone temperature control. The object content expected matches the ones received by "getTemperatureControlConfig2".

Synopsis

HTTP GET /json/zone/setTemperatureControlConfig2

Parameter

fields	Object with a collection of all fields that are to be changed	Optional
mode	Current Control Mode of the zone: "off", "control", "zoneFollower", "fixed", "manual"	Optional
targetTemperatures	Object with set point temperatures for each operation mode of the zone	Optional
fixedValues	Object fixed control values for each operation mode of the zone	Optional
controlMode	Object with the PID controller related parameters	Optional
zoneFollowerMode	Object with the zone follower related parameters	Optional
manualMode	Object with the manual mode parameter control value	Optional

Response

HTTP Status 200

```
ok true
```

Sample

```

GET /json/zone/setTemperatureControlConfig2?id=1237&mode=manual
{
  "ok": true,
}

GET /json/zone/setTemperatureControlConfig2?id=1237&targetTemperatures={"1": 23.5, "2": 22, "3": 19, "8" : 35}
{
  "ok": true,
}

GET /json/zone/setTemperatureControlConfig2?id=1237&fields={"mode": "control", "targetTemperatures":{"1": 23.5, "2": 22, "3": 19, "8" : 35}, "controlMode" : {"emergencyValue" : 42}}

```

```

{
  "ok": true,
}
GET /json/zone/setTemperatureControlConfig2?id=1237&fields={"mode":"manual", "manualMode":{"contolValue" : 45}}
{
  "ok": true,
}

```

3.7.8 Set Temperature Control State

Modify the internal state of the temperature control for a zone.

Synopsis

HTTP GET /json/zone/setTemperatureControlState

Notice Obsolete and has been removed in dSS Release 1.42.

3.7.9 Get Temperature Control Internals

Returns status information of the temperature control of a zone. Every controller attached to this reports its internal configuration and algorithm status data.

Synopsis

HTTP GET /json/zone/getTemperatureControlInternals

Parameter

None

Response

HTTP Status 200

result.DSUID	Object with the internal control parameters of this controller DSUID
--------------	----------------------------------------------------------------------

ControlMode	Control mode: 0=off; 1=pid-control; 2=zone-follower; 3=fixed-value; 4>manual
ControlState	Control state: 0=internal; 1=external; 2=exbackup; 3=emergency
CtrlTRecent	Current room temperature
CtrlTReference	Control temperature
CtrlTError	Control temperature error, in 0.025K
CtrlTErrorPrev	Previous control temperature error, in 0.025K
CtrlIntegral	Control current integral value
CtrlYp	Current control value proportional portion
CtrlYi	Current control value integral portion
CtrlYd	Current control value differential portion
CtrlY	Current control value
CtrlAntiWindUp	Currently the anti wind up condition is active

Sample

```
GET /json/zone/getTemperatureControlInternals?id=1237
```

```
{
  "ok":true,
  "result":
  {
    "3504175fe000000001000000006239100": {
      "ControlMode": 1,
      "ControlState": 0,
      "CtrlTRecent": 20.50,
      "CtrlTReference": 21.00,
      "CtrlTError": 0.55,
      "CtrlTErrorPrev": 0.50,
      "CtrlIntegral": 82,
      "CtrlYp": 3.55,
      "CtrlYi": 23.1,
      "CtrlYd": 0,
      "CtrlY": 27,
      "CtrlAntiWindUp": 0
    },
    ....
    "3504175fe00000000100000000714a300": {
      "ControlMode": 1,
      "ControlState": 2,
      "CtrlTRecent": 20.50,
      "CtrlTReference": 21.00,
      "CtrlTError": 0.55,
      "CtrlTErrorPrev": 0.50,
      "CtrlIntegral": 130,
      "CtrlYp": 3.55,
      "CtrlYi": 27.6,
      "CtrlYd": 0,
      "CtrlY": 29,
      "CtrlAntiWindUp": 0
    },
  }
}
```

4 Device

4.1 Common

Every `/json/device/` function uses a common selection scheme for the device to which the command refers to. Either the parameter `"dsuid"` or `"name"` must be given to identify the device.

Parameter	Description	Remarks
<code>dsuid</code>	Device dSUID String	Optional
<code>name</code>	Device Name	Optional
<code>category</code>	Request Category	Optional

A missing device identifier result in the following error message to be returned.

```
{ "ok": false, "message": "Need parameter name or dsuid to identify device" }
```

If a device identifier does not match any actually known device in the installation the following error message is returned.

```
{ "ok": false, "message": "Could not find device named 'Wandlampe am Eingang' }
```

The category parameter has an influence on how particular requests are treated, the goal is to prevent scene calls from automated scripts in certain situations. Currently supported categories are:

- manual
- timer
- algoirthm

A missing category parameter is currently treated as manual category, this compatibility will be removed in release 1.8.

4.2 Name

4.2.1 getName

Returns the user defined name of a device.

Synopsis

HTTP GET `/json/device/getName`

Parameter

None

Response

HTTP Status 200

<code>result.name</code>	identifier string for the device
--------------------------	----------------------------------

Sample

```
GET /json/device/getName?dsuid=3504175fe0000000000000000000000016c4f00
{
  "ok": true,
  "result":
  {
    "name": "App-Taster"
```

```
}  
}
```

4.2.2 setName

Sets the device name.

Synopsis

HTTP GET /json/device/setName

Parameter	Description	Remarks
newName	identifier string for the device	Mandatory

Parameter

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/device/setName?dsuid=3504175fe000000000000000000017ef300&newName="Wohnen"  
{  
  "ok":true  
}
```

4.2.3 getSpec

Retrieves device and product information.

Synopsis

HTTP GET /json/device/getSpec

Parameter

None

Response

HTTP Status 200

result.functionID	Function ID of the device
result.productID	Product ID of the device
result.revisionID	Revision ID of the device

Sample

```
GET /json/device/getName?dsuid=3504175fe000000000000000000016c4f00  
{  
  "ok": true,  
  "result": {  
    "functionID": 33027,  
    "productID": 1224,  
  }  
}
```

```
}
  "revisionID": 834
}
```

4.3 First seen

4.3.1 getFirstSeen

Returns the timestamp when the device was registered.

Synopsis

HTTP GET /json/device/getFirstSeen

Parameter

None

Response

HTTP Status 200

result.time	ISO8601 time when device was registered
-------------	-----------------------------------------

Sample

```
GET /json/device/getFirstSeen?dsuid=3504175fe000000000000000000000000016c4f00
{
  "result" :
  {
    "time" : "2010-10-01T10:42:13Z"
  },
  "ok":true
}
```

4.4 Groups

4.4.1 getGroups

Returns a list of groups the device is registered in.

Synopsis

HTTP GET /json/device/getGroups

Parameter

None

Response

HTTP Status 200

result.groups	array of groups of the device
---------------	-------------------------------

Sample


```

GET /json/device/getGroups
{
  "ok": true,
  "result": {
    "groups": [
      {
        "id": 3,
        "name": "blue"
      },
      {
        "id": 8,
        "name": "black"
      }
    ]
  }
}

```

4.5 Scene

4.5.1 callScene

Excutes the scene sceneNumber on a devices.

Synopsis

HTTP GET /json/device/callScene

Parameter

Parameter	Description	Remarks
sceneNumber	Numerical value	Mandatory
force	Boolean value, if set a forced scene call is issued	Optional

Response

HTTP Status 200

```
ok true
```

Sample

```

GET /json/device/callScene?dsuid=3504175fe0000000000000000000000016c4f00&sceneNumber=13
{
  "ok":true
}

```

4.5.2 saveScene

Tells the device to store the current output values as a default for the scene sceneNumber.

Synopsis

HTTP GET /json/device/saveScene

Parameter

Parameter	Description	Remarks
sceneNumber	Numerical value	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/saveScene?dsuid=3504175fe0000000000000000016c4f00&sceneNumber=5
{
  "ok":true
}
```

4.5.3 undoScene

Tells devices to restore the output values to the previous state if the current scene matches the sceneNumber.

Synopsis

HTTP GET /json/device/undoScene

Parameter

Parameter	Description	Remarks
sceneNumber	Numerical value	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/undoScene?dsuid=3504175fe0000000000000000016c4f00&sceneNumber=18
{
  "ok":true
}
```

4.5.4 turnOn

Tells devices to execute the scene MAX.

Synopsis

HTTP GET /json/device/turnOn

Parameter

None

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/turnOn?dsuid=3504175fe0000000000000000016c4f00
{
  "ok":true
}
```

4.5.5 turnOff

Tells devices to execute the scene MIN.

Synopsis

HTTP GET /json/device/turnOff

Parameter

None

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/turnOff?dsuid=3504175fe0000000000000000016c4f00
{
  "ok":true
}
```

4.5.6 increaseValue

Tells devices to execute the scene INC.

Synopsis

HTTP GET /json/device/increaseValue

Parameter

None

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/increaseValue?dsuid=3504175fe0000000000000000016c4f00
{
  "ok":true
}
```

4.5.7 decreaseValue

Tells devices to execute the scene DEC.

Synopsis

HTTP GET /json/device/decreaseValue

Parameter

None

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/decreaseValue?dsuid=3504175fe0000000000000000000000016c4f00
{
  "ok":true
}
```

4.6 Value

4.6.1 Set Value

Set the primary output value of a device to a given value.

Notice Setting output values directly bypasses the group state machine and is unrecommended.

Synopsis

HTTP GET /json/device/setValue

Parameter

Parameter	Description	Remarks
value	Numerical 8 bit value, in the range from 0 to 255	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setValue?dsuid=3504175fe0000000000000000000000016c4f00&value=127
{
  "ok":true
}
```

4.6.2 Set Output Value

Set a output channel value of a device to a given value. The available output parameter ranges and channels depend on the feature of the hardware components.

Notice Setting output values directly bypasses the group state machine and is unrecommended.

Synopsis

HTTP GET /json/device/setOutputValue

Parameter

Parameter	Description	Remarks
value	Numerical Value	Mandatory
offset	Output Data Offset	Mandatory

offset	Description	Product
2	shadeOpeningAngleOutside, 16 Bit	GR-KL220, GR-HKL230

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setOutputValue?dsuid=3504175fe000000000000000000000016c4f00&offset=2&value=5345
{
  "ok":true
}
```

4.6.3 Get Output Value

Get the current output channel status of a device. The available output channels depend on the feature of the hardware components.

Notice Getting output values directly from the device takes a noticeable amount of time. This request is subject of limitations in the systems certification rules.

Synopsis

HTTP GET /json/device/getOutputValue

Parameter

Parameter	Description	Remarks
offset	Output Data Offset	Mandatory
type	Named Parameter	Mandatory

Either offset or type parameter has to be given.

type	Description	Product
position	Target position, 16 Bit	GR-KL200, GR-KL210, GR-KL220, GR-KL230
angle	Target angle position, 8 Bit	GR-KL220, GR-KL230
positionCurrent	Current position when active, 16 Bit	GR-KL200, GR-KL210, GR-KL220, GR-KL230
pwmValue	Current PWM value, 8 Bit	BL-KM200, BL-SDS200
pwmPriorityMode	Current PWM priority value	BL-KM200, BL-SDS200
currentOutputMode	output operation mode when automatic selection is active	GE-KM300, GE-TKM300

Response

HTTP Status 200

result.offset	the given offset from the request
result.value	Numerical value of the selected output channel queried from the device

Sample

```
GET /json/device/getOutputValue?dsuid=3504175fe0000000000000000016c4f00&offset=0
{
  "ok": true, "result": { "offset": 0, "value": 5345 }
}
```

4.6.4 Get Scene Value

Retrieves the device value of the given scene.

Synopsis

HTTP GET /json/device/getSceneValue

Parameter

Parameter	Description	Remarks
sceneID	Numerical value	Mandatory

Response

HTTP Status 200

result.value	numerical output channel value of the device
result.angle	if available, angle value of the device
result.scenes	if available, a json object with a command field that is either a standard or custom device action

Sample

```
GET /json/device/getSceneValue?dsuid=3504175fe0000000000000017ef300&sceneID=72
{
  "ok":true,"result":{"value":65535,"angle":255}
}

GET /json/device/getSceneValue?dsuid=687ba4e345e75bd58093bf119f8a6c6700&sceneID=72
{
  "ok":true,"result":{"value":0}
}

GET /json/device/getSceneValue?dsuid=df6aa5bba4db5540c0fe55e3eb088be900&sceneID=72
"result": {
  "scenes": {
    "72": {
      "channels": null,
      "command": "std.stop",
      "dontCare": false,
      "effect": 1,
      "ignoreLocalPriority": true
    }
  }
},
"ok": true
}
```

4.6.5 Set Scene Value

Retrieves the device value of the given scene.

Synopsis

HTTP GET /json/device/setSceneValue

Parameter

Parameter	Description	Remarks
sceneID	Numerical value	Mandatory
value	Numerical value	Mandatory
angle	Numerical value, if applicable	Optional
command	String value, if applicable	Optional

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setSceneValue?dsuid=3504175fe0000000000000016c4f00&sceneID=72&value=26987
{
  "ok":true
}

GET /json/device/setSceneValue?dsuid=687ba4e345e75bd58093bf119f8a6c6700&sceneID=81&value=100&command=boilandcooldown
{
  "ok":true
}
```

4.6.6 Get Scene Mode

Reads the device configuration flags for a given sceneID. For details about the scene configuration see the ds-basics reference document.

Synopsis

HTTP GET /json/device/getSceneMode

Parameter

Parameter	Description	Remarks
sceneID	Scene number for which the configuration is requested	Mandatory

Response

HTTP Status 200

sceneID	Scene number which has been requested
dontCare	Don't Care Flag
localPrio	Local Prio Flag
specialMode	Special Mode Flag
flashMode	Flashing Mode Flag
ledconIndex	Index of the LED configuration register
dimTimeIndex	Index of the transition configuration register

Sample

```
GET /json/device/getSceneMode?dsuid=3504175fe0000000000000000000000016c4f00&sceneID=5
{
  "ok": true,
  "result":
  {
    "sceneID": 5,
    "dontCare": false,
    "localPrio": false,
    "specialMode": false,
    "flashMode": false,
    "ledconIndex": 1,
    "dimTimeIndex": 1
  }
}
```

4.6.7 Set Scene Mode

Sets the device configuration flags for a given sceneID. For details about the scene configuration see the ds-basics reference document.

Synopsis

HTTP GET /json/device/setSceneMode

Parameter

Synopsis

HTTP GET /json/device/getOutputChannelValue

Parameter

Parameter	Description	Remarks
channels	Semicolon separated list of channel names	Mandatory

Currently supported channels are listed below. For details please refer to the the dS-Basics document in the section Output Channel Types.

- brightness: light brightness
- hue: colored light hue
- saturation: colored light saturation
- colortemp: color temperature
- x: CIE color model x component
- y: CIE color model y component
- shadePositionOutside: shade position opening percentage for e.g. blinds and roller shutters
- shadePositionIndoor: shade position opening percentage for e.g. curtains
- shadeOpeningAngleOutside: shade position opening angle for e.g. lamellars
- shadeOpeningAngleIndoor: indoor shade position opening angle for e.g. lamellars
- transparency: transparency of e.g. a smart window
- airFlowIntensity: intensity of ventilation
- airFlowDirection: direction of air flow
- airFlapPosition: flap position
- airLouverPosition: louver position
- heatingPower: heating power and intensity
- coolingCapacity: cooling capacity and intensity
- audioVolume: audio loudness
- powerState: power status

Response

HTTP Status 200

result.channels	array of channels and their values
result.channels[x].channel	output channel name
result.channels[x].value	output channel value

Sample

```
GET /json/device/getOutputChannelValue?dsuid=3504175fe000000000000000000000000016c4f00&channels=brightness;saturation
{
  "ok":true,
  result: {
    channels: [
      { channel: "brightness", value: 50 },
      { channel: "saturation", value: 80 }
    ]
  }
}
```

4.6.10 Set Output Channel Value

Set the value of one or more output channels of the device.

Synopsis

HTTP GET /json/device/setOutputChannelValue

Parameter

Parameter	Description	Remarks
channelvalues	Semicolon separated list of channel names and their values	Mandatory
applyNow	Immediately apply the new values to the channe outputs	Optional, 1 (true) by default

See getOutputChannelValue description [4.6.9](#) for a list of output channel names and their value ranges.

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setOutputChannelValue?dsuid=3504175fe000000000000000000000000016c4f00&channelvalues=brightness=10;saturation=100&
  applyNow=1
{ "ok":true }
```

4.6.11 Get Output Channel Value v2

Retrieve the current output value of selected or all channels of the device.

Synopsis

HTTP GET /json/device/getOutputChannelValue2

Parameter

Parameter	Description	Remarks
channels	Semicolon separated list of channel names	Optional

If channels parameter is empty or omitted the call returns current values for all channels.

See `getOutputChannelValue` description [4.6.9](#) for a list of output channel names and their value ranges.

Response

HTTP Status 200

result.channels	array of channels and their values
result.channels[x].channel	output channel name
result.channels[x].value	output channel value

Sample

```
GET /json/device/getOutputChannelValue2?dsuid=5a11caa06212578280d826428d15c3d700&channels=brightness;saturation;hue
{
  "ok":true,
  result: {
    channels: {
      "brightness": { "value": 50, "automatic": false },
      "saturation": { "value": 80 },
      "hue": { "value": 0 }
    }
  }
}
```

4.6.12 Set Output Channel Value v2

Set the value of one or all output channels of the device.

Synopsis

HTTP GET `/json/device/setOutputChannelValue2`

Parameter

Parameter	Description	Remarks
channels	json object with channel names and their values	Mandatory
applyNow	Immediately apply the new values to the channels outputs	Optional, 1 (true) by default

The `channels` parameter json object has the same structure like returned by `getOutputChannelValue2`.

See `getOutputChannelValue` description [4.6.9](#) for a list of output channel names and their value ranges.

WARNING: Values passed to `setOutputChannelValue2` are not symmetric to values in `getOutputChannelValue2` response. Client is responsible to scale the values passed to `setOutputChannelValue2` to units used at dsm-api as specified in [ds-basics.pdf](#) section `Output Channel Types`.

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setOutputChannelValue2?dsuid=5a11caa06212578280d826428d15c3d700&channels={"brightness": {"value": 10,
    "automatic": false}, "saturation": {"value": 100}, "hue": {"value": 235}}
{"ok":true }
```

4.6.13 Get Output Channel Scene Value

Get scene value of one or more output channels.

Synopsis

HTTP GET `/json/device/getOutputChannelSceneValue`

Parameter

Parameter	Description	Remarks
channels	Semicolon separated list of channel names	Mandatory
sceneNumber	Number of the scene for which the values should be returned	Mandatory

See `getOutputChannelValue` description [4.6.9](#) for a list of output channel names and their value ranges.

Response

HTTP Status 200

result.sceneID	Scene number for which the values are returned
result.channels	array of channels and their values
result.channels[x].channel	output channel name
result.channels[x].value	output channel value for the requested scene

Sample

```
GET /json/device/getOutputChannelSceneValue?dsuid=3504175fe000000000000000000000000016c4f00&channels=brightness,saturation&
    sceneNumber=1
{
    "ok":true,
    result: {
```

```

    sceneID: 1,
    channels: [
      { channel: "brightness", value: 40 },
      { channel: "saturation", value: 20 }
    ]
  }
}

```

4.6.14 Set Output Channel Scene Value

Set scene value of one or more output channels.

Synopsis

HTTP GET /json/device/setOutputChannelSceneValue

Parameter

Parameter	Description	Remarks
channelvalues	Semicolon separated list of channel names and their values	Mandatory
sceneNumber	Number of the scene for which the values should be set	Mandatory

See getOutputChannelValue description [4.6.9](#) for a list of output channel names and their value ranges.

Response

HTTP Status 200

```
ok true
```

Sample

```

GET /json/device/setOutputChannelSceneValue?dsuid=3504175fe000000000000000000000000016c4f00&
channelvalues=brightness=10;saturation=100&sceneNumber=1
{
  "ok":true
}

```

4.6.15 Get Output Channel Scene Value v2

Reads the device configuration for a given sceneID and output channels.

Synopsis

HTTP GET /json/device/getOutputChannelSceneValue2

Parameter

Parameter	Description	Remarks
sceneNumber	Scene number for which the configuration is requested	Mandatory
channels	Semicolon separated list of channel names	Mandatory

See `getOutputChannelValue` description [4.6.9](#) for a list of output channel names and their value ranges.

Response

HTTP Status 200

sceneID	Scene number which has been requested
channels	Object with one JSON object per available channel type
channels.[x].value	Numeric channel value
channels.[x].dontCare	Don't Care Flag
channels.[x].automatic	Automatic Operation Flag

Sample

```
GET /json/device/getOutputChannelSceneValue2?dsuid=3504175fe0000000000000000016c4f00&sceneNumber=5&channels=airFlowIntensity;airLouverPosition
{
  "ok": true,
  "result":
  {
    "sceneID": 5,
    "channels": {
      "airFlowIntensity": { "value": 0, "dontCare": false, "automatic": true },
      "airLouverPosition": { "value": 50, "dontCare": false, "automatic": false }
    }
  }
}
```

4.6.16 Set Output Channel Scene Value v2

Sets the device configuration flags for a given `sceneID` and output channels.

Synopsis

HTTP GET `/json/device/setOutputChannelSceneValue2`

Parameter

Parameter	Description	Remarks
sceneNumber	Scene number which has been requested	Mandatory
channels	json object with channel names and their values	Mandatory

The `channels` parameter json object has the same structure like returned by `getOutputChannelSceneValue2`.

See `getOutputChannelValue` description [4.6.9](#) for a list of output channel names and their value ranges.

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/device/setOutputChannelSceneValue2?dsuid=3504175fe0000000000000000000000016c4f00&sceneNumber=5&channels={
  "airFlowIntensity" : {"dontCare": true}, "airLouverPosition" : {"value": 100, "automatic": true}}
{
  "ok": true
}
```

4.6.17 Get Output Channel Don't Care Flags

Get don't care flags for one or more output channels.

Notice `getOutputChannelDontCareFlag` is DEPRECATED. Please use [Get Output Channel Scene Mode](#) instead.

Synopsis

HTTP GET `/json/device/getOutputChannelDontCareFlags`

Parameter

Parameter	Description	Remarks
<code>sceneNumber</code>	Scene number for which the flag will be set	Mandatory

Response

HTTP Status 200

<code>result.channels</code>	array of channels and their values
<code>result.channels[x].channel</code>	output channel name
<code>result.channels[x].dontCare</code>	output channel "don't care" flag value

Sample

```
GET /json/device/getOutputChannelDontCareFlags?dsuid=3504175fe0000000000000000000000016c4f00&channels=brightness;saturation&
dontCare=1&sceneNumber=1
{
  "ok": true,
  result: {
    channels: [
      { channel: "brightness", dontCare: 0 },
      { channel: "saturation", dontCare: 1 }
    ]
  }
}
```


4.6.18 Set Output Channel Don't Care Flag

Set don't care flag for one or more output channels.

Notice `setOutputChannelDontCareFlag` is DEPRECATED. Please use [Set Output Channel Scene Mode](#) instead.

Synopsis

HTTP GET `/json/device/setOutputChannelDontCareFlag`

Parameter

Parameter	Description	Remarks
<code>channels</code>	Semicolon separated list of channel names	Mandatory
<code>dontCare</code>	Don't care flag value, boolean	Mandatory (0 or 1)
<code>sceneNumber</code>	Scene number for which the flag will be set	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setOutputChannelDontCareFlag?dsuid=3504175fe0000000000000000000000016c4f00&channels=brightness;saturation&dontCare=1&sceneNumber=1
{
  "ok":true
}
```

4.7 Configuration

4.7.1 setButtonID

Sets the button ID of a device. For details about the push button configuration see the ds-basics reference document.

Synopsis

HTTP GET `/json/device/setButtonID`

Parameter

Parameter	Description	Remarks
<code>buttonID</code>	Button number to set	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setButtonID?dsuid=3504175fe000000000000000000000016c4f00&buttonID=5
{
  "ok": true
}
```

4.7.2 setButtonInputMode

Sets the button input mode of a device. For details about the push button configuration see the ds-basics reference document.

Synopsis

HTTP GET /json/device/setButtonInputMode

Parameter

Parameter	Description	Remarks
modeID	Numerical value of the button mode to set	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setButtonInputMode?dsuid=3504175fe000000000000000000000016c4f00&modeID=0
{
  "ok": true
}
```

4.7.3 setOutputMode

Sets the output mode of a device.

Synopsis

HTTP GET /json/device/setOutputMode

Parameter

Parameter	Description	Remarks
modeID	Numerical value of the output mode to set	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setOutputMode?dsuid=3504175fe000000000000000000000016c4f00&modelID=0
{
  "ok": true
}
```

4.7.4 setJokerGroup

Sets the color group of a Joker device.

Synopsis

HTTP GET /json/device/setJokerGroup

Parameter

Parameter	Description	Remarks
groupID	Group number to set	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setJokerGroup?dsuid=3504175fe000000000000000000000016c4f00&groupID=2
{
  "ok": true
}
```

4.7.5 setButtonActiveGroup

Sets the user group of a push button device.

Synopsis

HTTP GET /json/device/setButtonActiveGroup

Parameter

Parameter	Description	Remarks
groupID	Group number to set	Mandatory, value range between 0 and 63,use 0xff to reset

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/device/setButtonActiveGroup?dsuid=3504175fe0000000000000000000000016c4f00&groupID=20
{
  "ok": true
}
```

4.7.6 getTransitionTime

Reads the device transition time configuration for a given register set. For details about the transition time configuration see the ds-basics reference document.

Synopsis

HTTP GET /json/device/getTransitionTime

Parameter

Parameter	Description	Remarks
dimtimeIndex	Index of the transition configuration register	Mandatory

Response

HTTP Status 200

dimtimeIndex	Index of the transition configuration register
up	Ramptime up in Milliseconds
down	Ramptime down in Milliseconds

Sample

```
GET /json/device/getTransitionTime?dsuid=3504175fe0000000000000000000000016c4f00&dimtimeIndex=2
{
  "ok": true,
  "result":
  {
    "dimtimeIndex": 2,
    "up": 600,
    "down": 55
  }
}
```

4.7.7 setTransitionTime

Sets the device transition time configuration for a given register set. For details about the transition time configuration see the ds-basics reference document.

Synopsis

HTTP GET /json/device/setTransitionTime

Parameter

Parameter	Description	Remarks
dimtimeIndex	Index of the transition configuration register	Mandatory
up	Ramptime up in Milliseconds	Mandatory
down	Ramptime down in Milliseconds	Mandatory

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/device/setTransitionTime?dsuid=3504175fe0000000000000000000000016c4f00&dimtimeIndex=2&up=600&down=600
{
  "ok": true
}
```

4.7.8 setConfig

Write a configuration value of a config class parameter to the device.

Notice Writing configuration parameters directly to the device may lead to malfunctions including complete failure of the whole device. Do not write parameters or values unless you are sure that the device supports it.

Synopsis

HTTP GET /json/device/setConfig

Parameter

Parameter	Description	Remarks
class	Configuration Class	Mandatory
index	Parameter Index	Mandatory
value	Parameter Value	Mandatory

Response

HTTP Status 200

class	the class parameter from the request
index	the index parameter from the request
value	parameter value

Sample

```
GET /json/device/setConfig?dsuid=3504175fe000000000000000000000000016c4f00&class=3&index=0&value=33
{
  "ok": true
}
```

4.7.9 getConfig

Reads a 8 bit parameter value of a config class from the device.

Notice Getting parameter values directly from the device takes a noticeable amount of time. This request is subject of limitations in the systems certification rules.

Synopsis

HTTP GET /json/device/getConfig

Parameter

Parameter	Description	Remarks
class	Configuration class	Mandatory
index	Parameter index	Mandatory

Response

HTTP Status 200

class	the class parameter from the request
index	the index parameter from the request
value	parameter value

Sample

```
GET /json/device/getConfig?dsuid=3504175fe000000000000000000000000016c4f00&class=1&index=2
{
  "ok": true,
  "result":
  {
    "class": 1,
    "index": 2,
    "value": 231
  }
}
```

4.7.10 getConfigWord

Reads a 16 bit parameter value of a config class from the device.

Notice Getting parameter values directly from the device takes a noticeable amount of time. This request is subject of limitations in the systems certification rules.

Synopsis

HTTP GET /json/device/getConfigWord

Parameter

Parameter	Description	Remarks
class	Configuration class	Mandatory
index	Parameter index, even	Mandatory

Response

HTTP Status 200

class	the class parameter from the request
index	the index parameter from the request
value	parameter value

Sample

```
GET /json/device/getConfigWord?dsuid=3504175fe0000000000000000000000016c4f00&class=3&index=2
{
  "ok": true,
  "result":
  {
    "class": 3,
    "index": 2,
    "value": 65280
  }
}
```

4.7.11 setCardinalDirection

Write the cardinal direction of the device.

Synopsis

HTTP GET /json/device/setCardinalDirection

Parameter

Parameter	Description	Remarks
direction	the cardinal direction of this device. Allowed values: none north north east east south east south south west west north west	

Response

HTTP Status 200

Sample

```
GET /json/device/setCardinalDirection?dsuid=3504175fe0000000000000000000000016c4f00&direction=south%20west
{
  "ok": true
}
```

4.7.12 getCardinalDirection

Read the configured cardinal direction of the device.

Synopsis

HTTP GET /json/device/getCardinalDirection

Response

HTTP Status 200

direction	the cardinal direction of this device. Allowed values: none north north east east south east south south west west north west
-----------	----------------------------------------------------------------------------------------------------------------------------------------------------------

Sample

```
GET /json/device/getCardinalDirection?dsuid=3504175fe0000000000000000000000016c4f00
{
  "ok": true,
  "result": {
    "direction": "south west"
  }
}
```

4.7.13 setWindProtectionClass

Write the wind protection class of the device.

Synopsis

HTTP GET /json/device/setWindProtectionClass

Parameter

Parameter	Description	Remarks
class	the wind protection class of this device.	

Response

HTTP Status 200

Sample

```
GET /json/device/setWindProtectionClass?dsuid=3504175fe0000000000000000016c4f00&class=4
{
  "ok": true
}
```

4.7.14 getWindProtectionClass

Read the the wid protection class of the device.

Synopsis

HTTP GET /json/device/getWindProtectionClass

Response

HTTP Status 200

class	the wind protection class of this device.
-------	-------------------------------------------

Sample

```
GET /json/device/getWindProtectionClass?dsuid=3504175fe0000000000000000016c4f00
{
  "ok": true,
  "result":
  {
    "class": 2
  }
}
```

4.7.15 setFloor

Write floor number where the device is installed.

Synopsis

HTTP GET /json/device/setFloor

Parameter

Parameter	Description	Remarks
floor	the floor number where the device is installed.	

Response

HTTP Status 200

Sample

```
GET /json/device/setFloor?dsuid=3504175fe000000000000000000000016c4f00&floor=14
{
  "ok": true
}
```

4.7.16 getFloor

Read floor number where the device is installed.

Synopsis

HTTP GET /json/device/getFloor

Response

HTTP Status 200

floor	the floor number where the device is installed.
-------	-------------------------------------------------

Sample

```
GET /json/device/getFloor?dsuid=3504175fe000000000000000000000016c4f00
{
  "ok": true,
  "result":
  {
    "floor": 14
  }
}
```

4.7.17 getMaxMotionTime

Reads the maximum motion time configuration of a shade device.

Synopsis

HTTP GET /json/device/getMaxMotionTime

Parameter

None

Response

HTTP Status 200

result.supported	boolean flag indicating if device supports this configuration
result.value	maximum motion time in seconds

Sample

```
GET /json/device/getMaxMotionTime?dsuid=3504175fe0000000000000000017bf600
{
  "result":
  {
    "supported": false,
    "value": 0
  },
  "ok": true
}
```

4.7.18 setMaxMotionTime

Configures the maximum motion time of a shade device.

Synopsis

HTTP GET /json/device/setMaxMotionTime

Parameter

Parameter	Description	Remarks
seconds	Maximum motion time in seconds where (0 < seconds < 655)	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setMaxMotionTime?dsuid=3504175fe0000000000000000016c4f00&seconds=10
{
  "ok": true
}
```

4.7.19 getOutputAfterImpulse

Reads configuration of an UMR device output after an impulse.

Synopsis

HTTP GET /json/device/getOutputAfterImpulse

Parameter

None

Response

HTTP Status 200

```
result.output | current output after impulse setting, can be "on", "off" or "retain"
```

Sample

```
HTTP GET /json/device/getOutputAfterImpulse?dsuid=302ed89f43f000000000ec00009478a00
{
  "result":
  {
    "output": "retain"
  },
  "ok": true
}
```

4.7.20 setOutputAfterImpulse

Configures UMR device output after an impulse.

Synopsis

HTTP GET /json/device/setOutputAfterImpulse

Parameter

Parameter	Description	Remarks
output	output setting: "on", "off" or "retain"	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/getOutputAfterImpulse?dsuid=302ed89f43f000000000ec00009478a00&output=off
{
  "ok": true
}
```

4.7.21 setVisibility

Configure TNY device visibility. Not allowed for "main" device.

Synopsis

HTTP GET /json/device/setVisibility

Parameter

Parameter	Description	Remarks
visibility	visibility setting: 0 or 1	Mandatory

Response

HTTP Status 200

result.action	one of "add", "none", "remove"
result.devices	array of devices to be processed

Sample

```
/json/device/setVisibility?dsuid=302ed89f43f0000000005400009f75d00&visibility=1  
  
{  
  "result":  
  {  
    "action": "none"  
  },  
  "ok": true  
}
```

4.7.22 setSupportedBasicScenes

Enables to provide a set of basic scenes supported by ventilation devices. Available scene numbers and their descriptions are: 0 - "Off", 5 - "Level1", 17 - "Level2", 18 - "Level3", 19 - "Level4", 36 - "Boost".

Synopsis

HTTP GET /json/device/setSupportedBasicScenes

Parameter

Parameter	Description	Remarks
value	Array of scene numbers	Mandatory

Response

HTTP Status 200

ok	true
----	------

Sample

```
/json/device/setSupportedBasicScenes?dsuid=302ed89f43f0000000005400009f75d00&value=[0,5,17,18,19,36]  
  
{  
  "ok": true  
}
```

4.7.23 getSupportedBasicScenes

Reads out a set of basic scenes supported by ventilation device.

Synopsis

HTTP GET /json/device/getSupportedBasicScenes

Parameter

None

Response

HTTP Status 200

value	Array of supported basic scenes
-------	---------------------------------

Sample

```

/json/device/getSupportedBasicScenes?dsuid=302ed89f43f00000000005400009f75d00
{
  "result":
  {
    "value": [0,17,18]
  },
  "ok": true
}

```

4.7.24 setIgnoreOperationLock

Sets ignore operation lock on a device

Synopsis

HTTP GET /json/device/setIgnoreOperationLock

Parameter

Parameter	Description	Remarks
lockStatus	lock status, boolean: 1 = lock ignored, 0 = lock observed	Mandatory (0 or 1)

Response

HTTP Status 200

ok	true
----	------

Sample

```

GET /json/device/setIgnoreOperationLock?dsuid=302ed89f43f0000000000ec00009478a00&lockStatus=1
{
  "ok": true
}

```

4.7.25 getIgnoreOperationLock

Gets ignore operation lock on a device

Synopsis

HTTP GET /json/device/getIgnoreOperationLock

Parameter

Response

HTTP Status 200

ignoreOperationLock	information whether ignore operation lock is set
---------------------	--------------------------------------------------

Sample

```
GET /json/device/getIgnoreOperationLock?dsuid=302ed89f43f0000000000ec00009478a00
{
  "ok": true
  "result": {
    "ignoreOperationlock": false
  }
}
```

4.8 Sensor

4.8.1 Get Sensor Value

Ready a sensor measurement from a device.

Synopsis

HTTP GET /json/device/getSensorValue

Parameter

Parameter	Description	Remarks
sensorIndex	Numerical value, in the range from 0 to 14	Mandatory

Response

HTTP Status 200

sensorIndex	the index parameter from the request
sensorValue	the actual measurement read from the device

Sample

```
GET /json/device/getSensorValue?dsuid=3504175fe0000000000000000016c4f00&sensorIndex=4
{
  "ok": true,
  "result": {
    "sensorIndex": 4,
    "sensorValue": 0
  }
}
```

4.8.2 Get Sensor Type

Ready the sensor type description from a device. For details about sensor types see the ds-basics reference document.

Synopsis

HTTP GET /json/device/getSensorType

Parameter

Parameter	Description	Remarks
sensorIndex	Numerical value, in the range from 0 to 14	Mandatory

Response

HTTP Status 200

sensorIndex	the index parameter from the request
sensorType	the sensor type read from the device

Sample

```
GET /json/device/getSensorType?dsuid=3504175fe0000000000000000000000016c4f00&sensorIndex=4
{
  "ok": true,
  "result": {
    "sensorIndex": 4,
    "sensorType": 6
  }
}
```

4.8.3 [getSensorEventTableEntry](#)

Reads the device event configuration for a given index. For details about the event table configuration see the ds-basics reference document.

Synopsis

HTTP GET /json/device/getSensorEventTableEntry

Parameter

Parameter	Description	Remarks
eventIndex	Index of the event configuration entry	Mandatory

Response

HTTP Status 200

eventIndex	Index of the event configuration register
eventName	User defined name of this event
sensorIndex	Sensor index on which this entry operates
action	Action value
value	Threshold value
test	Comparison operator
hysteresis	Hysteresis value
validity	Enabled Flag

Sample

```
GET /json/device/getSensorEventTableEntry?dsuid=3504175fe0000000000000000000000016c4f00&eventIndex=0
{
  "ok": true,
  "result": {
    "eventIndex": 0,
    "eventName": "",
    "sensorIndex": 2,
    "test": 2,
    "action": 0,
    "value": 35,
    "hysteresis": 0,
    "validity": 2
  }
}
```

4.8.4 setSensorEventTableEntry

Sets the device event configuration for a given index. For details about the event table configuration see the ds-basics reference document.

Synopsis

HTTP GET /json/device/setSensorEventTableEntry

Parameter

Parameter	Description	Remarks
eventIndex	Index of the event configuration register	Mandatory
eventName	User defined name of this event	Mandatory
sensorIndex	Sensor index on which this entry operates	Mandatory
action	Action value	Mandatory
value	Threshold value	Mandatory
test	Comparison operator	Mandatory
hysteresis	Hysteresis value	Mandatory
validity	Enabled Flag	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setSensorEventTableEntry?dsuid=3504175fe0000000000000000000000016c4f00&eventIndex=0&eventName="TV turned on"&
sensorIndex=2&test=2&action=0&value=50&hysteresis=25&validity=2
{
  "ok": true
}
```

4.9 Programming

4.9.1 Set Programming Mode

Enabled or disabled the programming mode on a device.

Synopsis

HTTP GET /json/device/setProgMode

Parameter

Parameter	Description	Remarks
mode	mode value, either enabled or disabled	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setProgMode?dsuid=3504175fe0000000000000000000000016c4f00&mode=disabled
{
  "ok": true
}
```

4.9.2 Add To Area

Modify the device scene table configuration and activate the area scene.

Synopsis

HTTP GET /json/device/addToArea

Parameter

Parameter	Description	Remarks
areaScene	either the area-on or area-off scenes	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/addToArea?dsuid=3504175fe0000000000000000000000016c4f00&areaScene=7
{
  "ok": true
}
```

4.9.3 Remove From Area

Modify the device scene table configuration and deactivate the area scene.

Synopsis

HTTP GET /json/device/removeFromArea

Parameter

Parameter	Description	Remarks
areaScene	either the area-on or area-off scenes	Mandatory

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/device/removeFromArea?dsuid=3504175fe0000000000000000016c4f00&areaScene=7
{
  "ok": true
}
```

4.9.4 Get Transmission Quality

Sends test commands to a device to evaluate the actual transmission quality.

Synopsis

HTTP GET /json/device/getTransmissionQuality

Parameter

None

Response

HTTP Status 200

upstream	a numerical value in the range of 0 to 62, 62 meaning best quality
downstream	a numerical value in the range 0 to 6, 0 meaning best quality

Sample

```
GET /json/device/getTransmissionQuality?dsuid=3504175fe0000000000000000016c4f00
{
  "ok": true,
  "result": {
    "upstream": 61,
    "downstream": 0
  }
}
```

4.10 Heating and valve actuators

4.10.1 setHeatingGroup

Sets the standard color group of a heating actuator. Some actuators support operation with different connected hardware equipment, therefore the terminal blocks support operation in different zone groups, for example in heating, cooling or ventilation.

Synopsis

HTTP GET /json/device/setHeatingGroup

Parameter

Parameter	Description	Remarks
groupID	New group Id	Mandatory

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/device/setHeatingGroup?dsuid=3504175fe000000000000000000000016be700&groupID=48
{
  "ok": true
}
```

4.10.2 getValvePwmState

Reads the device status of a valve PWM actuator.

Synopsis

HTTP GET /json/device/getValvePwmState

Parameter

None

Response

HTTP Status 200

pwmValue	Current PWM control value in percent
pwmPriorityMode	Current operating state value

Sample

```
GET /json/device/getValvePwmState?dsuid=3504175fe000000000000000000000016be700
{
  "ok": true,
  "result":
  {
    "pwmValue": 60,
    "pwmPriorityMode": 0
  }
}
```

4.10.3 getValvePwmMode

Reads the device configuration of a valve PWM actuator.

Synopsis

HTTP GET /json/device/getValvePwmMode

Parameter

None

Response

HTTP Status 200

pwmPeriod	Length of PWM period in seconds
pwmMinX	Minimum set point or threshold
pwmMaxX	Maximum set point or threshold
pwmMinY	Minimum output value at min set point
pwmMaxY	Maximum output value at max set point
pwmOffset	Set point offset

Sample

```
GET /json/device/getValvePwmMode?dsuid=3504175fe0000000000000000000016be700
{
  "ok": true,
  "result":
  {
    "pwmPeriod": 900,
    "pwmMinX": 0,
    "pwmMaxX": 10,
    "pwmMinY": 12,
    "pwmMaxY": 28,
    "pwmOffset": -20
  }
}
```

4.10.4 setValvePwmMode

Sets the device configuration of a valve PWM actuator.

Synopsis

HTTP GET /json/device/setValvePwmMode

Parameter

Parameter	Description	Remarks
pwmPeriod	Length of PWM period in seconds, 0 .. 64k	Optional
pwmMinX	Minimum set point or threshold, 0 .. 255	Optional
pwmMaxX	Maximum set point or threshold, 0 .. 255	Optional
pwmMinY	Minimum output value at min set point, 0 .. 255	Optional
pwmMaxY	Maximum output value at max set point, 0 .. 255	Optional
pwmOffset	Set point offset, -128 .. 127	Optional

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setValvePwmMode?dsuid=3504175fe000000000000000000000016be700&pwmMaxX=80&pwmOffset=-20
{
  "ok": true
}
```

4.10.5 getValveControlMode

Reads the device configuration of a valve PWM actuator.

Synopsis

HTTP GET /json/device/getValveControlMode

Parameter

None

Response

HTTP Status 200

ctrlNONC	Configure normally closed (false) or normally open (true) output behavior
ctrlClipMaxHigher	PWM value over maximum control value to 100 percent
ctrlClipMinLower	PWM lower than maximum control value to 0 percent
ctrlClipMinZero	Control value of zero maps to 0 percent PWM

Sample

```
GET /json/device/getValveControlMode?dsuid=3504175fe000000000000000000000016be700
{
  "ok": true,
  "result":
  {
    "ctrlNONC": true,
    "ctrlClipMaxHigher": false,
    "ctrlClipMinLower": false,
    "ctrlClipMinZero": false
  }
}
```

4.10.6 setValveControlMode

Sets the device configuration of a valve PWM actuator.

Synopsis

HTTP GET /json/device/setValveControlMode

Parameter

Parameter	Description	Remarks
ctrlClipMinZero	Control value of zero maps to 0 percent PWM	Optional
ctrlClipMinLower	PWM lower than minimum forces control value to 0 percent	Optional
ctrlClipMaxHigher	PWM value over maximum forces control value to 100 percent	Optional
ctrlNONC	Configure normally open or normally closed output behavior	Optional

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/device/setValveControlMode?dsuid=3504175fe000000000000000000000000016be700&ctrlNONC=false
{
  "ok": true
}
```

4.10.7 getValveTimerMode

Reads the timer device configuration settings of a valve PWM actuator.

Synopsis

HTTP GET /json/device/getValveTimerMode

Parameter

None

Response

HTTP Status 200

valveProtectionTimer	Duration of the valve protection period in seconds
emergencyValue	Fixed output value in percent in emergency mode
emergencyTimer	Duration in seconds until emergency mode is activated

Sample

```
GET /json/device/getValveTimerMode?dsuid=3504175fe000000000000000000000000016be700
{
  "ok": true,
  "result":
  {
    "valveProtectionTimer": 0,
    "emergencyValue": 75,
    "emergencyTimer": 7200
  }
}
```

```
}  
}
```

4.10.8 setValveTimerMode

Sets the timer device configuration of a valve PWM actuator.

Synopsis

HTTP GET /json/device/setValveTimerMode

Parameter

Parameter	Description	Remarks
valveProtectionTimer	Duration of the valve protection period in seconds	Optional
emergencyValue	Fixed output value in percent in emergency mode	Optional
emergencyTimer	Duration in seconds until emergency mode is activated	Optional

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/setValveTimerMode?dsuid=3504175fe000000000000000000000000016be700&valveProtectionTimer=600  
{  
  "ok": true  
}
```

4.11 Single Device Info

The Single Device Info section refers to the device description data that is available only for selected devices. Please read the Device Description section of the system interfaces documentation.

If the device is not a Single Device with device description data the following error message will be returned:

```
{  
  "ok": false,  
  "message": "Device does not support action configuration"  
}
```

4.11.1 Get Info Static

Returns the static device description data of a device. This data is available in a database on the digitalSTROM-Server and is not fetched from the device itself.

Synopsis

HTTP GET /json/device/getInfoStatic

Parameter

Parameter	Description	Remarks
lang	Locale Code based on Language_Region pattern, e.g. en_EN, de_DE	Mandatory

Response

HTTP Status 200

spec	common device description object
stateDescriptions	list of state description objects
eventDescriptions	list of event description objects
propertyDescriptions	list of property description objects
sensorDescriptions	list of sensor description objects
actionDescriptions	list of action description objects
standardActions	list of standard action description objects

spec

```
"spec": {
  "descriptionId": {
    "title": "Translated title for key descriptionId",
    "value": "Value of the key descriptionId",
    "tags": "string value, semi-colon separated list of attributes"
  }
}
```

Common descriptionId's are:

"name"	User given name of the dSDevice
"dsDeviceGTIN"	GTIN of the dSDevice
"model"	Product Name
"modelVersion"	Product/Model Revision
"vendorName"	Vendor/Maker
"vendorId"	Vendor ID, numerical number
"hardwareGuid"	Instance ID of the hardware, e.g. S/N, MAC Address, SGTIN
"hardwareModelGuid"	Model ID of the hardware, e.g. GTIN of the Native Device
"class"	dS Class/Profile Name, e.g. "Water Kettle", "Dishwasher"
"classId"	dS Class/Profile ID
"classVersion"	Revision Number of the supported class/profile

stateDescriptions

```
"stateTechnicalName": {
  "title": "Translated title for this state object",
  "options": { list of "OptionId": "OptionValue" pairs }
  "tags": "optional string value, semi-colon separated list of attributes"
}
```

Example:

```

"operation": {
  "title": "Betriebszustand",
  "options": {
    "idle": "Bereitschaft",
    "active": "Aktiv",
    "error": "Fehler"
  },
  "tags": null
}

```

eventDescriptions

```

"eventTechnicalName": {
  "title": "Translated title for this event object",
  "accessLevel": optional enum of
    "normal" (default) // Shown everywhere
    "advanced" // Shown within the UI if the 'show more' events option was selected
    "service" // Not shown within the UIs at all but might be used by bots and other services.
}

```

propertyDescriptions

```

"parameterTechnicalName": {
  "title": "Translated name of this parameter",
  "type": "data type of the parameter value: numeric, enumeration, string",
  "tags": "string value, semi-colon separated list of attributes"
}

```

Additional optional fields for type numeric:

min	minimum value
max	maximum value
resolution	minimum step size
siunit	unit string, http://www.ebyte.it/library/educards/siunits/TablesOfSiUnitsAndPrefixes.html , e.g. "second"
default	a default value of the property

Additional optional fields for type enumeration:

options	json object with a list of "OptionId": "OptionValue" pairs
default	a default value of the property

Additional optional fields for type text:

max	maximum length of the string value
default	a default value of the property

Following fields are defined for the tags attribute:

readonly	parameter value can only be read and not written
invisible	parameter shall not be shown and must be hidden in the UI
overview	state or property shall be shown on the overview tab in the UI, order/position can be given with "number"
settings	state or property shall be shown on the settings tab in the UI, order/position can be given with "number"

The "type" field string might have a postfix that indicates the characteristic of the value. This can be used for rendering the data field in user interfaces.

Following postfix descriptions are defined for the type attribute:

numeric.timeOfDay	hh:mm or am/pm depending on the region setting, does not include time zone
numeric.duration	hh:mm:ss or hh:mm, depending on unit and resolution
numeric.boolean	true/false, displayed as checkbox

Example:

```

"waterhardness": {
  "title": "Wasserh\u00e4rte",
  "type": "numeric",
  "min": 0,
  "max": 6,
  "resolution": 0.1,
  "default": 2.1,
  "tags": "readonly"
}

```

sensorDescriptions

```

"sensorTechnicalName": {
  "title": "Translated name of this measurement",
  "type": "dataTypeOfTheMeasurementValue:numeric,enumeration,string",
  "tags": "stringValue,semi-colon separated list of attributes"
}

```

The sensor object is represented by the same extended fields then property objects, depending on the "type" field.

Additional mandatory fields for sensor objects are:

dsType	device sensor type id number as defined by dS
dsIndex	device index of the source, necessary to address in queries

actionDescriptions

```

"actionDescriptions": {
  "actionId1": {
    "title": "Translated label for actionId1",
    "params": list of {propertyDescriptions}
  },
  ...,
  "actionIdN": {
    "title": "Translated label for actionIdN",
    "params": list of {propertyDescriptions}
  }
},

```

standardActions

```

"standardActions": {
  "std.Action1": {
    "action": "reference to the base action description",
    "title": "Translated name for std.Action1",
    "params": { list of "ParameterName": ParameterValue, ... }
  },

```

```

....
"std.ActionN": {
  "action": "reference[]to[]the[]base[]action[]description",
  "title": "Translated[]name[]for[]std.ActionN",
  "params": { list of "ParameterName": ParameterValue, ...}
}
}

```

Sample

```

GET /json/device/getInfoStatic?dsuid=687ba4e345e75bd58093bf119f8a6c6700&lang=de_DE
{
  "result": {
    "spec": {
      "dsDeviceGTIN": {
        "title": "dS[]Device[]GTIN",
        "tags": "settings:5",
        "value": "7640156791945"
      },
      "hardwareGuid": {
        "title": "Artikel[]Kennzeichnung",
        "tags": "settings:4",
        "value": "MAC[]5C:CF:7F:11:F8:B8"
      },
      "hardwareModelGuid": {
        "title": "Produkt[]Kennzeichnung",
        "tags": "invisible",
        "value": "smartermodel:ikettle2"
      },
      "model": {
        "title": "Modell",
        "tags": "overview:2;settings:2",
        "value": "iKettle[]2"
      },
      "modelVersion": {
        "title": "Modellvariante",
        "tags": "invisible",
        "value": "19"
      },
      "name": {
        "title": "Name",
        "tags": "overview:1;settings:1",
        "value": "Wasserkocher"
      },
      "notes": {
        "title": "Bemerkungen",
        "tags": "overview:4",
        "value": "Bitte[]prüfen[]Sie[]mit[]der[]'Smarter[]Smartphone[]App,[]ob[]die[]iKettle[]Firmware[]auf[]dem[]aktuellsten[]Stand[]ist!"
      },
      "vendorId": {
        "title": "Hersteller[]Kennung",
        "tags": "invisible",
        "value": "vendorname:Smarter[]Applications[]Ltd."
      },
      "vendorName": {
        "title": "Hersteller",
        "tags": "overview:3;settings:3",
        "value": "Smarter[]Applications[]Ltd."
      },
      "class": {
        "title": "Geräteklasse",
        "tags": "invisible",
        "value": ""
      },
      "classVersion": {
        "title": "Geräteklassen[]Version",
        "tags": "invisible",
        "value": ""
      }
    },
    "stateDescriptions": {
      "operation": {
        "title": "Betriebsmodus",
        "tags": "overview",
        "options": {

```

```

        "cooldown": "Abkühlen",
        "heating": "Aufheizen",
        "keepwarm": "Warmhalten",
        "ready": "Bereit",
        "removed": "Abgehoben"
    }
},
"eventDescriptions": {
    "KettleAttached": {
        "title": "Kocher␣aufgesetzt"
    },
    "BoilingStarted": {
        "title": "Aufheizen␣gestartet"
    },
    "KeepWarm": {
        "title": "Warmhalten␣gestartet"
    },
    "BabycoolingStarted": {
        "title": "Aufheizen␣beendet,␣␣auf␣Zieltemperatur␣abkühlen"
    },
    "BoilingFinished": {
        "title": "Aufheizen␣beendet"
    },
    "KettleReleased": {
        "title": "Kocher␣abgehoben"
    },
    "BabycoolingFinished": {
        "title": "Abkühlen␣beendet,␣Zieltemperatur␣erreicht"
    },
    "KeepWarmFinished": {
        "title": "Warmhalten␣beendet"
    },
    "BoilingAborted": {
        "title": "Aufheizen␣abgebrochen,␣Taste␣betätigt"
    },
    "BabycoolingAborted": {
        "title": "Abkühlen␣abgebrochen,␣Taste␣betätigt"
    },
    "KeepwarmAborted": {
        "title": "Warmhalten␣abgebrochen,␣Taste␣betätigt"
    },
    "KeepWarmAfterBoiling": {
        "title": "Aufheizen␣beendet,␣warmhalten"
    },
    "KeepWarmAfterBabycooling": {
        "title": "Abkühlen␣beendet,␣warmhalten"
    },
    "BoilingAbortedAndKettleReleased": {
        "title": "Aufheizen␣abgebrochen,␣Kocher␣abgehoben"
    },
    "BabyCoolingAbortedAndKettleReleased": {
        "title": "Abkühlen␣abgebrochen,␣Kocher␣abgehoben"
    },
    "KeepWarmAbortedAndKettleReleased": {
        "title": "Warmhalten␣abgebrochen,␣Kocher␣abgehoben"
    }
},
"propertyDescriptions": {
    "currentTemperature": {
        "title": "Wassertemperatur",
        "tags": "readonly;overview",
        "type": "numeric",
        "min": "0",
        "max": "100",
        "resolution": "1",
        "siunit": "celsius",
        "default": "0"
    },
    "waterLevel": {
        "title": "Füllstand",
        "tags": "readonly;overview",
        "type": "numeric",
        "min": "0",
        "max": "2.0",
        "resolution": "0.2",
        "siunit": "liter",
        "default": "0"
    }
}

```

```

    },
    "defaulttemperature": {
      "title": "Temperatur␣Aufheizen",
      "tags": "settings",
      "type": "numeric",
      "min": "0",
      "max": "100",
      "resolution": "1",
      "siunit": "celsius",
      "default": "100"
    },
    "defaultcooldowntemperature": {
      "title": "Temperatur␣Abkochen␣und␣Abk\u00fchlen",
      "tags": "invisible",
      "type": "numeric",
      "min": "0",
      "max": "100",
      "resolution": "1",
      "siunit": "celsius",
      "default": "80"
    },
    "defaultkeepwarmtime": {
      "title": "Warmhaltezeit",
      "tags": "settings",
      "type": "numeric",
      "min": "0",
      "max": "30",
      "resolution": "1",
      "siunit": "min",
      "default": "15"
    }
  },
  "sensorDescriptions": {
    "waterQuantity": {
      "title": "Wassermenge",
      "tags": "readonly;overview",
      "type": "numeric",
      "min": "0",
      "max": "8",
      "resolution": "0.1",
      "siunit": "liter",
      "dsType": 68,
      "dsIndex": 3
    }
  },
  "actionDescriptions": {
    "boilandcooldown": {
      "title": "Abkochen␣und␣Abk\u00fchlen",
      "params": {
        "keepwarmtime": {
          "title": "Warmhaltedauer",
          "tags": "",
          "type": "numeric",
          "min": "0",
          "max": "30",
          "resolution": "1",
          "siunit": "min",
          "default": "30"
        },
        "temperature": {
          "title": "Zieltemperatur",
          "tags": "",
          "type": "numeric",
          "min": "20",
          "max": "100",
          "resolution": "1",
          "siunit": "celsius",
          "default": "50"
        }
      }
    },
    "heat": {
      "title": "Aufheizen",
      "params": {
        "keepwarmtime": {
          "title": "Warmhaltedauer",
          "tags": "",
          "type": "numeric",

```

```

        "min": "0",
        "max": "30",
        "resolution": "1",
        "siunit": "min",
        "default": "30"
    },
    "temperature": {
        "title": "Zieltemperatur",
        "tags": "",
        "type": "numeric",
        "min": "20",
        "max": "100",
        "resolution": "1",
        "siunit": "celsius",
        "default": "100"
    }
},
"stop": {
    "title": "Abschalten",
    "params": {}
}
},
"standardActions": {
    "std.boilandcooldown": {
        "title": "Abkochen und abk\u00fchlen",
        "action": "boilandcooldown",
        "params": {
            "temperature": "40"
        }
    },
    "std.heat": {
        "title": "Aufheizen",
        "action": "heat",
        "params": {}
    },
    "std.stop": {
        "title": "Abschalten",
        "action": "stop",
        "params": {}
    }
}
},
"ok": true
}

```

4.11.2 Get Info Operational

Returns the current value for states and properties.

Synopsis

HTTP GET /json/device/getInfoOperational

Parameter

Parameter	Description	Remarks
lang	Locale Code based on Language_Region pattern, e.g. en_EN, de_DE	Mandatory

Response

HTTP Status 200

states	list of state value objects
properties	list of property value objects

States and Property names are corresponding to the response of the static descriptions. The static response contains translations and other meta information.

states

```
"states": {
  "stateTechnicalName1": {
    "value": "stateOptionValue"
  },
  ...,
  "stateTechnicalNameN": {
    "value": "stateOptionValue"
  }
},
```

properties

```
"properties": {
  "propertyTechnicalName1": {
    "value": number
  },
  ...,
  "propertyTechnicalNameN": {
    "value": number
  }
},
```

measurements

```
"sensors": {
  "sensorTechnicalName1": {
    "value": number
  },
  ...,
  "sensorTechnicalNameN": {
    "value": number
  }
},
```

Sample

```
GET /json/device/getInfoOperational?dsuid=687ba4e345e75bd58093bf119f8a6c6700&lang=de_DE
{
  "result": {
    "states": {
      "operation": {
        "age": 1.756285,
        "changed": 45452.848575,
        "value": "ready"
      }
    },
    "properties": {
      "currentTemperature": 25,
      "defaultkeepwarmtime": 30,
      "defaulttemperature": 100,
      "waterLevel": 1.8620689655172413
    },
    "sensors": {
      "waterQuantity": 8.6
    }
  },
  "ok": true
}
```


4.11.3 Get Info Custom

Returns the custom actions defined by the user or define. The custom actions are configurable and are available in addition to the standard actions.

Synopsis

HTTP GET /json/device/getInfoCustom

Parameter

Parameter	Description	Remarks
lang	Locale Code based on Language_Region pattern, e.g. en_EN, de_DE	Mandatory

Response

HTTP Status 200

customActions	list of action description objects
---------------	------------------------------------

The custom action name is given by the user. Each custom action is based on a defined and known actionDescription of the device.

customActions

```
"customActions": {
  "custom.753151": {
    "action": "reference[]to[]the[]base[]action[]description",
    "title": "User[]given[]name[]for[]custom.753151",
    "params": { list of "ParameterName": ParameterValue, ...}
  },
  .....,
  "custom.143937": {
    "action": "reference[]to[]the[]base[]action[]description",
    "title": "User[]given[]name[]for[]custom.143937",
    "params": { list of "ParameterName": ParameterValue, ...}
  }
}
```

Sample

```
GET /json/device/getInfoCustom?dsuid=687ba4e345e75bd58093bf119f8a6c6700&lang=de_DE
{
  "result": {
    "customActions": {
      "custom.582620628227b": {
        "action": "std.boilandcooldown",
        "params": {
          "keepwarmtime": 30,
          "temperature": 70
        },
        "title": "Früchtetee"
      },
      "custom.5826208698525": {
        "action": "std.heat",
        "params": {
          "keepwarmtime": 0,
          "temperature": 42
        },
        "title": "Lauwarmes[]Wasser"
      },
      "custom.582620a92e329": {
        "action": "std.heat",
```

```

    "params": {
      "keepwarmtime": 15,
      "temperature": 66
    },
    "title": "Spülwasser␣aufwärmen"
  },
  "custom.58404582ef972": {
    "action": "std.boilandcooldown",
    "params": {
      "keepwarmtime": 0,
      "temperature": 100
    },
    "title": "Wasser␣abkochen"
  }
}
},
"ok": true
}

```

4.11.4 Get Info

getInfo is a method that combines static, operational and custom information in one call. With the given filter parameter the caller can select which response fields he likes to have in the response.

Synopsis

HTTP GET /json/device/getInfo

Parameter

Parameter	Description	Remarks
lang	Locale Code based on Language_Region pattern, e.g. en_EN, de_DE	Mandatory
filter	string with a comma separated list of response objects	Optional

The filter parameter accepts the following options: spec, standardActions, customActions, stateDesc, propertyDesc, sensorDesc, actionDesc, eventDesc, operational.

If filter parameter is omitted or empty the full set of response objects is returned.

Response

HTTP Status 200

The response is a combination of the getInfoStatic, getInfoCustom and getInfoOperational response. Please refer to the descriptions above.

4.11.5 Set Property

This method allows to change property values that are part of a getInfo Property Description.

Synopsis

HTTP GET /json/device/setProperty

Parameter

Parameter	Description	Remarks
dsuid	dSUID of the device	Mandatory
id	property identifier	Mandatory
value	new value of the property	Mandatory

The id parameter corresponds to the property technical name from getInfo response.

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/device/setProperty?dsuid=687ba4e345e75bd58093bf119f8a6c6700&id=defaultkeepwarmtime&value=90.5
{
  "ok": true
}
```

4.11.6 Set Custom Action

This method allows to create or replace custom actions of a device.

Synopsis

HTTP GET /json/device/setCustomAction

Parameter

Parameter	Description	Remarks
id	unique custom action identifier, must have prefix "custom."	Mandatory
title	user given name or title of this action	Mandatory
action	reference to basic action description identifier	Mandatory
params	json object with a list of property values: "PropertyName": "PropertyValue" , ...	Mandatory

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/device/setCustomAction?dsuid=687ba4e345e75bd58093bf119f8a6c6700&id=custom.123456&title=Lauwarmes Wasser&
action=std.heat&params={"temperature":40}
{
  "ok": true
}
```

4.11.7 Call Action

Excutes the action id on a device.

Synopsis

HTTP GET /json/device/callAction

Parameter

Parameter	Description	Remarks
id	standard or custom action identifier	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/device/callAction?dsuid=df6aa5bba4db5540c0fe55e3eb088be900&id=std.stop
{
  "ok":true
}
```

4.11.8 Get Apartment Scenes

Retrieves the list of device values for all supported apartment scenes (sceneID 64 and above).

Synopsis

HTTP GET /json/device/getSceneValue

Parameter

None

Response

HTTP Status 200

```
result.scenes a list of json objects per sceneID
```

Sample

```
GET /json/device/getApartmentScenes?dsuid=df6aa5bba4db5540c0fe55e3eb088be900
"result": {
  "scenes": {
    "64": {
      "channels": null,
      "command": "std.stop",
      "dontCare": false,
      "effect": 1,
      "ignoreLocalPriority": true
    },
    .....,
    "92": {
      "channels": null,
```

```
    "command": "std.stop",
    "dontCare": false,
    "effect": 1,
    "ignoreLocalPriority": true
  }
},
"ok": true
}
```

4.12 Test presence

4.12.1 testPresence

Tests the device presence.

Synopsis

HTTP GET /json/device/testPresence

Parameter

None

Response

HTTP Status 200

result.active	bool - is the device active / inactive?
---------------	-----------------------------------------

Sample

```
GET /json/device/getName?dsuid=3504175fe0000000000000000000000016c4f00
{
  "ok":true,
  "result" :
  {
    "active" : true
  }
}
```

4.13 Button Usage

Most devices have a button input, but not all of them are connected to push-buttons. A user interface may only display button inputs which were used before, ignoring all the terminal blocks where no buttons are connected to. A button is "used" if it got pushed before, "auto_unused" if it got not pushed yet, or "manual_unused" if a user set it into this state.

4.13.1 getButtonUsage

Returns information about button input usage of the device, only supported on devices that have button inputs.

Synopsis

HTTP GET /json/device/getButtonUsage

Parameter

None

Response

HTTP Status 200

result.buttonUsage	button usage setting (used, auto_unused, manual_unused, unsupported)
--------------------	----------------------------------------------------------------------

Sample

```
GET /json/device/getButtonUsage?dsuid=3504175fe0000000000000016c4f00
{
  "ok":true,
  "result" :
  {
    "buttonUsage" : "used"
  }
}
```

4.13.2 setButtonUsage

Sets the button usage state, only supports devices that have buttons.

Synopsis

HTTP GET /json/device/setButtonUsage

Parameter	Description	Remarks
usage	usage identifier (used, auto_unused, manual_unused)	Mandatory

Parameter

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/device/setButtonUsage?dsuid=3504175fe0000000000000016c4f00&usage=manual_unused
{
  "ok":true
}
```

5 Circuit

5.1 Common

Every /json/circuit/ function uses a common selection scheme for the connected infrastructure component to which the command refers to. This component can be either a connected digitalSTROM-Meter or an IP Device Connector VDC.

The parameter "dsuid" must be given to identify the component which is a string value of the dSUID.

Parameter	Description	Remarks
dsuid	dSUID Number of the infrastructure component	Mandatory

A missing dsuid identifier result in the following error message to be returned.

```
{
  "ok": false,
  "message": "Missing parameter dsuid"
}
```

If a dSUID identifier does not match any actually known component in the installation the following error message is returned.

```
{
  "ok": false,
  "message": "Could not find dSMeter with given dsuid"
}
```

5.2 Name

5.2.1 getName

Returns the user defined name of the zone.

Synopsis

HTTP GET /json/circuit/getName

Parameter

None

Response

HTTP Status 200

name	identifier string for the digitalSTROM-Meter
------	----------------------------------------------

Sample

```
GET /json/circuit/getName?dsuid=3504175fe00000000000010000004d500
{
  "ok": true,
  "result": {
    "name": "Wohnen/Flur/Eingang"
  }
}
```

5.2.2 setName

Sets the zone name.

Synopsis

HTTP GET /json/circuit/setName

Parameter	Description	Remarks
newName	identifier string for the digitalSTROM-Meter	Mandatory

Parameter

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/circuit/setName?dsuid=3504175fe000000000000010000004d500&newName="Wohnen"  
{  
  "ok":true  
}
```

5.3 Energy Meter

5.3.1 getConsumption

Returns the current measurement of the power consumption on this circuit.

Synopsis

HTTP GET /json/circuit/getConsumption

Parameter

None

Response

HTTP Status 200

consumption	Current power consumption [W]
-------------	-------------------------------

Sample

```
GET /json/circuit/getConsumption?dsuid=3504175fe000000000000010000004d500  
{  
  "ok": true,  
  "result": {  
    "consumption": 725  
  }  
}
```

5.3.2 getEnergyMeterValue

Returns the current measurement of the power consumption on this circuit.

Synopsis

HTTP GET /json/circuit/getEnergyMeterValue

Parameter

None

Response

HTTP Status 200

meterValue	Energy Meter Value [Ws]
------------	-------------------------

Sample

```
GET /json/circuit/getEnergyMeterValue?dsuid=3504175fe000000000000010000004d500
{
  "ok": true,
  "result": {
    "meterValue": 1438467
  }
}
```

5.3.3 getMeteringConfiguration

Returns the metering parameters for this digitalSTROM-Meter.

Synopsis

HTTP GET /json/circuit/getMeteringConfiguration

Parameter

None

Response

HTTP Status 200

category	identifier string for the digitalSTROM-Meter metering category
scaleFactor	effective numeric factor as multiplier for the measured energy

Sample

```
GET /json/circuit/getName?dsuid=3504175fe000000000000010000004d500
{
  "ok": true,
  "result": {
    "category": "producer",
    "scaleFactor": 1
  }
}
```

5.3.4 setMeteringConfiguration

Sets the metering parameters for this digitalSTROM-Meter.

Synopsis

HTTP GET /json/circuit/setMeteringConfiguration

Parameter

Parameter	Description	Remarks
category	identifier string for the digitalSTROM-Meter metering category	
scaleFactor	effective numeric factor as multiplier for the measured energy	

ok	true
----	------

Response

HTTP Status 200

Sample

```
GET /json/circuit/setMeteringConfiguration?dsuid=3504175fe00000000000010000004d500&category="producer"&scaleFactor=3
{
  "ok":true
}
```

5.4 Configuration

5.4.1 learnIn

Enable and allow to register new devices and establish new connections. Typically the registration of new devices is a teach-in process that requires action e.g. button press on the physical device itself to pair with a new peer.

Synopsis

HTTP GET /json/circuit/learnIn

Parameter

Parameter	Description	Remarks
timeout	time in seconds until the lean in process is disabled again	mandantory
params	device specific parameters	optional, key/value pairs encoded json

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/circuit/learnIn?dsuid=0963dd2d722d5dc6c0ecb2aa7465465600&timeout=30
{
  "ok":true,
}
```

5.4.2 learnOut

Revert the teach-in process and deregister devices.

Synopsis

HTTP GET /json/circuit/learnOut

Parameter	Description	Remarks
timeout	time in seconds until the lean out process is disabled again	mandantory
params	device specific parameters	optional, key/value pairs encoded json

Parameter

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/circuit/learnOut?dsuid=0963dd2d722d5dc6c0ecb2aa7465465600&timeout=30
{
  "ok":true
}
```

5.4.3 firmwareCheck

Test for firmware verification and availability of updates.

Synopsis

HTTP GET /json/circuit/firmwareCheck

Parameter None

Response

HTTP Status 200

ok	true
status	firmware status code: "ok", "error", "update"

ok	up-to-date, no firmware update available
error	the firmware status could not be checked, e.g. due to missing connectivity
update	an update is available and ready for installation

Sample

```
GET /json/circuit/firmwareCheck?dsuid=0963dd2d722d5dc6c0ecb2aa7465465600
{
  "ok":true,
  "status": "update"
}
```

5.4.4 firmwareUpdate

Start the firmware upgrade process. This process is running autonomously. Typically the device will restart and register again.

Synopsis

HTTP GET /json/circuit/firmwareUpdate

Parameter	Description	Remarks
clearsettings	request to reset all data to factory defaults	optional

Parameter

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/circuit/firmwareUpdate?dsuid=0963dd2d722d5dc6c0ecb2aa7465465600&clearsettings=true
{
  "ok":true
}
```

5.4.5 storeAccessToken

Receives authentication token and related data for a device.

Synopsis

HTTP GET /json/circuit/storeAccessToken

Parameter	Description	Remarks
authData	authorization data string, content is device specific	mandantory
authScope	scope, device specific	optional

Parameter

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/circuit/storeAccessToken?dsuid=0963dd2d722d5dc6c0ecb2aa7465465600&authData={"access_token":
"42EC27D0AD0616334CB670C29211ABF1693C71666C6158960DD51DFFB4B18150", "expires": 86400}&
authScope=user@domain.com,ReadData
{
  "ok":true
}
```

5.5 Reregister devices

5.5.1 reregisterDevices

Reregisters devices on selected controller.

Synopsis

HTTP GET /json/circuit/reregisterDevices

Parameter

Parameter	Description	Remarks
dsuid	dSUID of the device	Mandatory

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/circuit/reregisterDevices?dsuid=3504175fe0000000000000000017ef300
{
  "ok":true,
}
```

6 Structure

6.1 Zone

6.1.1 addZone

Adds a zone with the given Id. The zone is added to the digitalSTROM-Server data model only and initially does not have any devices associated.

Synopsis

HTTP GET /json/structure/addZone

Parameter

Parameter	Description
zoneID	unique numerical identifier for the new zone
floorID	ID of the floor in which the zone will be created. If not specified, the default floor will be assigned

Response

HTTP Status 200

ok	true
result.zoneID	numeric identifier for the new zone

Sample

```
GET /json/structure/addZone?zoneID=1&floorID=34
{
  "ok":true,
  "result":
  {
    "zoneID":1
  }
}
```

6.1.2 removeZone

Removes the zone with the give Id from the installation. A zone can only be removed if it has no associated devices.

Synopsis

HTTP GET /json/structure/removeZone

Parameter	Description	Remarks
zoneID	unique numerical identifier	Mandatory

Parameter

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/structure/removeZone?zoneID=1234
{
  "ok":true
}
```

6.1.3 floorAddZone

Associates a zone with a new floor. The zone is automatically removed from the old floor.

Synopsis

HTTP GET /json/structure/floorAddZone

Parameter

Parameter	Description	Remarks
zoneID	ID of the zone to move	Mandatory
floorID	ID for the new floor	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/structure/floorAddZone?zoneID=1234&floorID=23
{
  "ok":true
}
```

6.2 Floor

6.2.1 addFloor

Adds a floor.

Synopsis

HTTP GET /json/structure/addFloor

Response

HTTP Status 200

ok	true
result.floorID	numeric identifier for the new floor
result.orderID	numeric identifier for sorting the floor in UIs

Sample

```
GET /json/structure/addFloor
{
  "ok":true,
  "result":
  {
    "floorID":1,
    "orderID":3
  }
}
```

6.2.2 removeFloor

Removes the floor with the give Id from the installation. A floor can only be removed if it has no associated zones.

Synopsis

HTTP GET /json/structure/removeFloor

Parameter	Description	Remarks
floorID	unique numerical identifier	Mandatory

Parameter

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/structure/removeFloor?floorID=1234
{
  "ok":true
}
```

6.2.3 setFloorName

Sets the floor name.

Synopsis

HTTP GET /json/zone/setFloorName

Parameter	Description	Remarks
name	identifier string for the floor	Mandatory

Parameter

Response

HTTP Status 200


```
ok true
```

Sample

```
GET /json/zone/setFloorName?id=1237&name="1st floor"
{
  "ok":true
}
```

6.3 Group

6.3.1 addGroup

Adds a user group to the zone with the given Id.

Synopsis

HTTP GET /json/structure/addGroup

Parameter

Parameter	Description	Remarks
zoneID	identifier of the zone where the group has to be created	Mandatory
groupID	numerical identifier for the new group	Mandatory if groupAutoSelect is not given
groupAutoSelect	flag to let the system find a free group id	Mandatory if groupID is not given
groupColor	application state machine selector for the new group, default is none	Optional
groupName	name for the new group	Optional

Response

HTTP Status 200

result.groupID	numeric identifier for the new group
result.zoneID	numeric identifier for the zone
result.groupName	string, name of the new group
result.groupColor	numeric identifier, color of the new group

Sample

```
GET /json/structure/addGroup?zoneID=1234&groupAutoSelect=global&groupColor=5&groupName=test
{
  "ok":true,
  "result":
  {
    "groupID":41,
```

```

    "zoneID":1234,
    "groupName":"test",
    "groupColor":5
  }
}

```

6.3.2 removeGroup

Removes a user group to the zone with the given Id.

Synopsis

HTTP GET /json/structure/removeGroup

Parameter

Parameter	Description	Remarks
zoneID	unique numerical identifier for the zone	Mandatory
groupID	numerical identifier for the group	Mandatory

Response

HTTP Status 200

```

ok true

```

Sample

```

GET /json/structure/removeGroup?zoneID=1234&groupID=42
{
  "ok":true
}

```

6.3.3 groupSetName

Rename a group.

Synopsis

HTTP GET /json/structure/groupSetName

Parameter

Parameter	Description	Remarks
zoneID	unique numerical identifier for the zone	Mandatory
groupID	numerical identifier for the group	Mandatory
newName	string, new name for the group	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/structure/groupSetName?zoneID=1234&groupID=42&newName=test
{
  "ok":true
}
```

6.3.4 groupSetColor

Change application type of the zone user group or apartment user application.

Synopsis

HTTP GET /json/structure/groupSetColor

Parameter

Parameter	Description	Remarks
zoneID	unique numerical identifier for the zone	Mandatory
groupID	numerical identifier for the group	Mandatory
newColor	numerical identifier of the application type the group	Mandatory

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/structure/groupSetColor?zoneID=1234&groupID=42&newColor=4
{
  "ok":true
}
```

6.3.5 groupSetConfiguration

Set application specific attributes for a group. The following attributes are supported:

Attribute	Application	Remarks
activeBasicScenes	Ventilation, Recirculation	Configure the available levels for ventilation groups

Synopsis

HTTP GET /json/structure/groupSetConfiguration

Parameter

Parameter	Description	Remarks
zoneID	unique numerical identifier for the zone	Mandatory
groupID	numerical identifier for the group	Mandatory
configuration	string with encoded json object defining the attributes	Mandatory

ok	true
----	------

Response

HTTP Status 200

Sample

```
GET /json/structure/groupSetConfiguration?zoneID=1234&groupID=10&configuration={"activeBasicScenes":[0,5,17]}
{
  "ok":true
}
```

6.3.6 groupGetConfiguration

Get application specific attributes for a group.

Synopsis

HTTP GET /json/structure/groupSetConfiguration

Parameter

Parameter	Description	Remarks
zoneID	unique numerical identifier for the zone	Mandatory
groupID	numerical identifier for the group	Mandatory
configuration	string with encoded json object defining the attributes	Mandatory

Response

HTTP Status 200

ok	true
result.activeBasicScenes	array of basic ventilation scene numbers, e.g. levels (optional)

Sample

```
GET /json/structure/groupGetConfiguration?zoneID=1234&groupID=10
{
  "result": {
    "activeBasicScenes": [
      0,
      5,
      17,
      18,
      19,
      36
    ]
  }
}
```

```
},  
  "ok": true  
}
```

6.4 Cluster

6.4.1 addCluster

Adds a cluster with the given name and color and returns the automatically chosen Id.

Synopsis

HTTP GET /json/structure/addCluster

Parameter

Parameter	Description	Remarks
color	application state machine selector for the new cluster	Mandatory
name	name for the new group	Mandatory

Response

HTTP Status 200

result.clusterID	numeric identifier for the new cluster
result.name	string, name of the new cluster
result.color	numeric identifier, color of the new cluster

Sample

```
GET /json/structure/addCluster?color=2&name=test  
{  
  "ok": true,  
  "result":  
  {  
    "clusterID": 34,  
    "name": "test",  
    "color": 2  
  }  
}
```

6.4.2 removeCluster

Removes a cluster with the given Id.

Synopsis

HTTP GET /json/structure/removeCluster

Parameter

Parameter	Description	Remarks
clusterID	numerical identifier for the cluster	Mandatory

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/structure/removeCluster?clusterID=34
{
  "ok":true
}
```

6.4.3 clusterSetName

Rename a cluster.

Synopsis

HTTP GET /json/structure/clusterSetName

Parameter

Parameter	Description	Remarks
clusterID	numerical identifier for the cluster	Mandatory
newName	string, new name for the cluster	Mandatory

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/structure/clusterSetName?clusterID=18&newName=test
{
  "ok":true
}
```

6.4.4 clusterSetColor

Change color of the cluster.

Synopsis

HTTP GET /json/structure/clusterSetColor

Parameter

Parameter	Description	Remarks
clusterID	numerical identifier for the cluster	Mandatory
newColor	numerical identifier of the color for the cluster	Mandatory

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/structure/clusterSetColor?clusterID=18&newColor=4
{
  "ok":true
}
```

6.4.5 clusterSetConfigLock

Locks or unlocks the configuration of the cluster. If locked changes of the target application and devices are not allowed.

Synopsis

HTTP GET /json/structure/clusterSetConfigLock

Parameter

Parameter	Description	Remarks
clusterID	numerical identifier for the cluster	Mandatory
lock	numerical value: 0 = unlocked, 1 = locked	Mandatory

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/structure/clusterSetConfigLock?clusterID=18&lock=1
{
  "ok":true
}
```

6.5 Device

6.5.1 zoneAddDevice

Associates a device with a new zone. A device is automatically removed from the old zone. Only active devices can be moved to a new zone because the zone configuration has to be synchronized with the device itself.

Synopsis

HTTP GET /json/structure/zoneAddDevice

Parameter

Parameter	Description	Remarks
dsuid of the device to move	Mandatory	
zone	unique numerical identifier for the new zone	Mandatory

Response

HTTP Status 200

movedDevices	array of devices which have been moved
--------------	----------------------------------------

In the case of a failure various different error messages may occur.

Sample

```
GET /json/structure/zoneAddDevice?dsuid=3504175fe00000000000000000000585400&zone=1
{
  ok: true
  result: {
    movedDevices: [
      {
        dsUID: "3504175fe00000000000000000000585400"
        name: ""
        functionID: 4144
        productRevision: 788
        productID: 1234
        hwInfo: "GE-TKM210"
        meterDSUID: "3504175fe00000000000010000004d900"
        busID: 241
        zoneID: 1
        isPresent: true
        lastDiscovered: "2012-11-22 10:35:05"
        firstSeen: "2012-11-19 14:34:02"
        inactiveSince: "1970-01-01 00:00:00"
        outputMode: 16
        buttonID: 12
        buttonActiveGroup: 1
        buttonInputMode: 0
        buttonInputIndex: 0
        buttonInputCount: 1
        groups: [
          "1"
        ]
      }
    ]
  }
}
```

6.5.2 removeDevice

Removes a device from the data model. Only inactive devices can be removed.

Synopsis

HTTP GET /json/structure/removeDevice

Parameter

Parameter	Description	Remarks
dsuid of the device to be removed	Mandatory	

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/structure/removeDevice?dsuid=3504175fe0000000000000000000000585400
{
  ok: false
  message: "Cannot remove present device"
}
```

6.5.3 groupAddDevice

Adds a device to the user group. Only active devices can be added to additional groups.

Synopsis

HTTP GET /json/structure/groupAddDevice

Parameter

Parameter	Description	Remarks
dsuid	DSUID of the device	Mandatory
groupID	unique numerical group identifier	Mandatory

Response

HTTP Status 200

ok	true
action	either update or none
devices	list of devices that have been changed, only existing if action is "update"

Sample

```
GET /json/structure/groupAddDevice?dsuid=3504175fe0000000000000000000000585400&groupID=42
{
  ok: true,
  result: {
    action: update,
    devices: [
      {
        dSUID: "3504175fe0000000000000000000000585400",
        name: "",
        functionID: 4144,
        productRevision: 788,
        productID: 1234,
        hwInfo: "GE-TKM210",
        meterDSUID: "3504175fe000000000000010000004d900",
        busID: 241,
      }
    ]
  }
}
```

```

    zoneID: 1,
    isPresent: true,
    lastDiscovered: "2012-11-22 10:35:05",
    firstSeen: "2012-11-19 14:34:02",
    inactiveSince: "1970-01-01 00:00:00",
    outputMode: 16,
    buttonID: 12,
    buttonActiveGroup: 1,
    buttonInputMode: 0,
    buttonInputIndex: 0,
    buttonInputCount: 1,
    groups: [
      "1", "42"
    ]
  }
}
}
}

```

6.5.4 groupRemoveDevice

Removes a device from the user group. Only active devices can be removed from groups.

Synopsis

HTTP GET /json/structure/groupRemoveDevice

Parameter

Parameter	Description	Remarks
dsuid	DSUID of the device	Mandatory
groupID	unique numerical group identifier	Mandatory

Response

HTTP Status 200

ok	true
action	either update or none
devices	list of devices that have been changed, only existing if action is "update"

Sample

```

GET /json/structure/groupRemoveDevice?dsuid=3504175fe00000000000000000000000585400&groupID=42
{
  ok: true,
  result: {
    action: update,
    devices: [
      {
        dSUID: "3504175fe00000000000000000000000585400",
        name: "",
        functionID: 4144,
        productRevision: 788,
        productID: 1234,
        hwInfo: "GE-TKM210",
        meterDSUID: "3504175fe000000000000000000000004d900",
        busID: 241,
        zoneID: 1,
        isPresent: true,
        lastDiscovered: "2012-11-22 10:35:05",
        firstSeen: "2012-11-19 14:34:02",

```

```

    inactiveSince: "1970-01-01T01:00:00",
    outputMode: 16,
    buttonID: 12,
    buttonActiveGroup: 1,
    buttonInputMode: 0,
    buttonInputIndex: 0,
    buttonInputCount: 1,
    groups: [
      "1"
    ]
  }
]
}
}
}

```

6.5.5 clusterAddDevice

Adds a device to the cluster. Only active devices can be added to additional cluster.

Synopsis

HTTP GET /json/structure/clusterAddDevice

Parameter

Parameter	Description	Remarks
dsuid	DSUID of the device	Mandatory
clusterID	unique numerical cluster identifier	Mandatory

Response

HTTP Status 200

ok	true
action	either update or none
devices	list of devices that have been changed, only existing if action is "update"

Sample

```

GET /json/structure/clusterAddDevice?dsuid=3504175fe00000000000000000000000585400&groupID=16
{
  ok: true,
  result: {
    action: update,
    devices: [
      {
        dSUID: "3504175fe00000000000000000000000585400",
        name: "",
        functionID: 4144,
        productRevision: 788,
        productID: 1234,
        hwInfo: "GE-TKM210",
        meterDSUID: "3504175fe000000000000000000000004d900",
        busID: 241,
        zoneID: 1,
        isPresent: true,
        lastDiscovered: "2012-11-22T10:35:05",
        firstSeen: "2012-11-19T14:34:02",
        inactiveSince: "1970-01-01T01:00:00",
        outputMode: 16,
        buttonID: 12,
        buttonActiveGroup: 1,

```



```
}  
  }  
    }  
      ]  
        "1"
```

7 Event and State

7.1 Raise Event

7.1.1 raise

Raises an event and appends it to the digitalSTROM-Server event queue. Details of the digitalSTROM-Server event processing can be found in the system-interfaces document.

Notice System events should be treated as reserved and must not be raised by external applications. In this term system events are events which originate from the digitalSTROM system lower layers.

Synopsis

HTTP GET /json/event/raise

Parameter

Parameter	Description	Remarks
name	identifier string for event	Mandatory
parameter	list of key=value pairs, seperated with semicolons	Optional

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/event/raise?name=highlevevent&parameter=id=1026;value=0;index=1
{
  "ok":true
}
```

7.2 Event Subscription

7.2.1 subscribe

Subscribe to an event with the given name and registers the callers subscriptionId. A unique subscriptionId can be selected by the subscriber. It is possible to subscribe to several events reusing the same subscriptionId.

Synopsis

HTTP GET /json/event/subscribe

Parameter	Description	Remarks
name	identifier string for the event	Mandatory
subscriptionID	numerical unique value	Mandatory

Parameter

Response

HTTP Status 200

```
ok true
```

Sample

```
GET /json/event/subscribe?name=deviceSensorEvent&subscriptionID=42
{
  "ok":true
}
```

7.2.2 unsubscribe

Unsubscribes for the previously registered events by giving the event name and the unique subscriptionId.

Synopsis

HTTP GET /json/event/unsubscribe

Parameter	Description	Remarks
name	identifier string for the event	Mandatory
subscriptionID	numerical unique value	Mandatory

Parameter

Response

HTTP Status 200

```
ok true
```

If there is no registered session for the given event name the following error message is returned.

```
{
  ok: false
  message: "Event" callScene" is not subscribed in this session"
}
```

If the subscriptionId is unknown to the digitalSTROM-Server the following error message is returned.

```
{
  ok: false
  message: "Token not found!"
}
```

Sample

```
GET /json/event/unsubscribe?name=callScene&subscriptionID=42
{
  ok: true
}
```

7.2.3 get

Get event and context information for an event subscription. All events subscribed with the given Id will be handled by this call. An optional timeout value in milliseconds can be specified and will block the call until either an event or the timeout occurs. If the timeout value is zero or missing the call will not timeout.

Synopsis

HTTP GET /json/event/get

Parameter	Description	Remarks
subscriptionID	numerical unique value	Mandatory
timeout	numerical value, timeout in milli seconds	Optional

Parameter

Response

HTTP Status 200

events	array of events
--------	-----------------

Sample

```
GET /json/event/get?subscriptionID=42&timeout=60000
{
  ok: true
  result: {
    events: []
  }
}
```

```
GET /json/event/get?subscriptionID=42&timeout=60000
{
  ok: true
  result: {
    events: [
      {
        name: "callScene"
        properties: {
          groupID: "1"
          sceneID: "8"
          zoneID: "1241"
          originDeviceID: "3504175fe000000000000000000585400"
        }
      }
    ]
  }
}
```

7.3 State

7.3.1 set

Sets the value of a system state. Details of digitalSTROM-Server system states can be found in the system-interfaces document.

Notice Only a subset of the system states can be changed by this method. Many systems states reflect a physical status of an e.g. input line and cannot be modified.

Synopsis

HTTP GET /json/state/set

Parameter

Parameter	Description	Remarks
name	identifier for the state	mandatory
value	new value	mandatory
addon	specify the owner of the state	optional

Response

HTTP Status 200

ok	true
----	------

Sample

```
GET /json/state/set?name=heating_system&value=off
{
  "ok":true
}
GET /json/state/set?addon=system-addon-user-defined-states&name=1484843926&value=active
{
  "ok":true
}
```

8 Metering

8.1 Metering

8.1.1 getResolution

Returns a list of time-series metering data resolutions stored on the digitalSTROM-Server.

Synopsis

HTTP GET /json/metering/getResolutions

Parameter

None

Response

HTTP Status 200

Parameter	Description
ok	boolean result of the call
result.resolutions	a list of supported resolutions
result.resolutions[...].resolution	step size in thei resolution in seconds

Sample

```
GET /json/metering/getResolutions
{
  "ok": true,
  "result": {
    "resolutions": [
      {
        "resolution": 1
      },
      {
        "resolution": 60
      },
      {
        "resolution": 900
      },
      {
        "resolution": 86400
      },
      {
        "resolution": 604800
      },
      {
        "resolution": 2592000
      }
    ]
  }
}
```

8.1.2 getSeries

Returns a list of all metering series stored on the digitalSTROM-Server.

Three types of series are available:

energy An energy meter counter. (cumulative)

energyDelta The total energy consumed during the previous time slot. (derivative)

consumption The average power used during the previous time slot.

Synopsis

HTTP GET /json/metering/getSeries

Parameter

None

Response

HTTP Status 200

Parameter	Description
ok	boolean result of the call
result.series	a list of available time series
result.values[...].dSUID	dSUID of the digitalSTROM-Meter for this series
result.series[...].type	the series type

Sample

```
GET /json/metering/getSeries
{
  "ok": true,
  "result": {
    "series": [
      {
        "dsuid": "3504175fe0000000000000000000000016c4f00",
        "type": "energy"
      },
      {
        "dsuid": "3504175fe0000000000000000000000016c4d00",
        "type": "energyDelta"
      },
      {
        "dsuid": "3504175fe0000000000000000000000016c4e00",
        "type": "power"
      }
    ]
  }
}
```

8.1.3 getValues

Returns a time series of metering values with the specified properties.

All times are integers that represent UNIX timestamps (seconds since 1970-01-01).

The (optional) window selection parameters can be used in different combinations. Only two of the three options can be used together in a call. The following table details the available combinations:

startTime return all available values starting at startTime until now.

endTime return all available values from the oldest available until endTime.

valueCount return the valueCount newest values

startTime and valueCount return valueCount values starting from startTime.

endTime and valueCount return valueCount values ending at endTime

startTime and endTime return the values between startTime and endTime.

Synopsis

HTTP GET /json/metering/getValues

Parameter

Parameter	Description	Remarks
dsuid	request the data for this digitalSTROM-Meter	Mandatory
type	series type (according to the getSeries call)	Mandatory
resolution	series resolution (the digitalSTROM-Server will adjust the resolution to the closest multiple of the available resolutions according to the getResolutions call)	Mandatory
unit	(only relevant for types "energy" and "energyDelta") unit of the returned metering values. Options are "Wh" and "Ws". Defaults to "Wh".	Optional
startTime	start time (UNIX timestamp)	Optional
endTime	ent time (UNIX timestamp)	Optional
valueCount	number of values (UNIX timestamp)	Optional

Response

HTTP Status 200

Parameter	Description
ok	boolean result of the call
result.meterID	dSUID of the digitalSTROM-Meter
result.type	same as Request
result.resolution	actual resolution of the data, might differ from the requested resolution if it was not available.
result.values	array of time-value pairs

Sample

```
GET /json/metering/getValues?dsuid=3504175fe00000000100000063a00&type=energy&resolution=60&unit=Ws&valueCount=5
{
  "ok": true,
  "result": {
    "meterID": "3504175fe00000000100000063a00",
    "type": "energy",
    "unit": "Ws",
    "resolution": "60",
    "values": [
      [
        1352906040,
        47562600
      ],
      [
        1352906100,
        47562600
      ],
      [
        1352906160,
        47562600
      ],
      [
        1352906220,
        47562600
      ]
    ]
  }
}
```

```

    1352906280,
    47562600
  ]
}
}
}

```

8.1.4 getAggregatedValues

Returns the sum of time series of metering values with the specified properties for all digitalSTROM-Meter's.

Synopsis

HTTP GET /json/metering/getAggregatedValues

Parameter

Parameter	Description	Remarks
dsuid	request the data as sum for a list of digitalSTROM-Meter, argument can be a single dsuid, a list of dsuids given in ".meters(dsuid1, dsuid2)" syntax, or the meta command .meters(all)	Mandatory
type	series type (according to the getSeries call)	Mandatory
resolution	series resolution (the digitalSTROM-Server will adjust the resolution to the closest multiple of the available resolutions according to the getResolutions call)	Mandatory
unit	(only relevant for types "energy" and "energyDelta") unit of the returned metering values. Options are "Wh" and "Ws". Defaults to "Wh".	Optional
startTime	start time (UNIX timestamp)	Optional
endTime	ent time (UNIX timestamp)	Optional
valueCount	number of values (UNIX timestamp)	Optional

Response

HTTP Status 200

Parameter	Description
ok	boolean result of the call
result.meterID	array of dSUID's of the requested digitalSTROM-Meter's
result.type	same as Request
result.resolution	actual resolution of the data, might differ from the requested resolution if it was not available.
result.values	array of summed up result values according to the given time period

Sample

```

GET /json/metering/getAggregatedValues?
  dsuid=.meters(303505d7f80000000000002c00000379c00,3504175fe00000000000010000004d900)&
  type=energy&resolution=60&unit=Ws&valueCount=5
{
  "result": {
    "meterID": [

```

```

        "303505d7f8000000000002c00000379c00",
        "3504175fe00000000000010000004d900"
    ],
    "type": "energy",
    "unit": "Ws",
    "resolution": "60",
    "values": [
        [
            1448980080,
            9559790.0
        ], [
            1448980140,
            9560116.0
        ], [
            1448980200,
            9560438.0
        ], [
            1448980260,
            9560760.0
        ]
    ]
},
"ok": true
}

```

8.1.5 getLatest

Returns the latest available metering values.

Synopsis

HTTP GET /json/metering/getLatest

Parameter

Parameter	Description	Remarks
from	the dSUID of the requested digitalSTROM-Meters. It uses a Set-Syntax: ".meters(dsuid1,dsuid2,...)" and ".meters(all)"	Mandatory
type	series type (according to the getSeries call)	Mandatory
unit	(only relevant for types "energy" and "energyDelta") unit of the returned metering values. Options are "Wh" and "Ws". Defaults to "Wh".	Optional

Response

HTTP Status 200

Parameter	Description
ok	boolean result of the call
result.values	array of results
result.values[...].dsuid	dSUID of the digitalSTROM-Meter
result.values[...].value	the latest metering value
result.values[...].date	date and time when the latest metering value was recorded

Sample

```

GET /json/metering/getLatest?from=.meters(3504175fe00000100000063a,3504175fe0000010000008c4)&type=energy&unit=Ws
{
  "ok": true,
  "result": {

```

```

    "values": [
      {
        "dsuid": "3504175fe0000000000000000016c4f00",
        "value": 49414887,
        "date": "2012-11-19T13:49:41"
      },
      {
        "dsuid": "3504175fe0000000000000000016c4e00",
        "value": 151215631,
        "date": "2012-11-19T10:29:29"
      }
    ]
  }
}

```

8.1.6 getAggregatedLatest

Returns the sum of latest metering values for all digitalSTROM-Meter's.

Synopsis

HTTP GET /json/metering/getAggregatedLatest

Parameter

Parameter	Description	Remarks
from	the dSUID of the requested digitalSTROM-Meters. It uses a Set-Syntax: ".meters(dsuid1,dsuid2,...)" and ".meters(all)"	Mandatory
type	series type (according to the getSeries call)	Mandatory
unit	(only relevant for types "energy" and "energyDelta") unit of the returned metering values. Options are "Wh" and "Ws". Defaults to "Wh".	Optional

Response

HTTP Status 200

Parameter	Description
ok	boolean result of the call
result.values	array of results
result.values[...].dSUID	dSUID of the digitalSTROM-Meter
result.values[...].value	the summed up latest metering value
result.values[...].date	date and time when the latest metering value was recorded

Sample

```

GET /json/metering/getAggregatedLatest?
  from=.meters{303505d7f80000000000002c00000379c00,3504175fe00000000000010000004d900}&
  type=energy&unit=Ws
{
  "result": {
    "values": [
      {
        "dSUID": [
          "303505d7f80000000000002c00000379c00",
          "3504175fe00000000000010000004d900"
        ],
        "value": 4214,
        "date": "2015-12-01T15:38:16"
      }
    ]
  }
}

```

```
]
},
"ok": true
}
```


9 System

9.1 System Information

9.1.1 version

Returns the version of the digitalSTROM Server software.

Synopsis

HTTP GET /json/system/version

Parameter

None

Response

HTTP Status 200

version	the dSS application version
distroVersion	the host platform firmware release (since v1.10)
Hardware	the host platform hardware identifier (since v1.10)
Revision	the host platform hardware revision number (since v1.10)
Serial	the host platform hardware serial number (since v1.10)
Ethernet	the host platform IEEE Mac address (since v1.10)
MachineID	the host system unique id (since v1.10)
Kernel	the host system Linux kernel release string (since v1.10)

Sample

```
GET /json/system/version
{
  ok: true,
  result:
  {
    version: "dSSv1.31.1[git:2acc82fe90f273a788fb573b07419f29f369a02a][oebuild@builder]",
    distroVersion: "Angstrom2010.4-devel-20111031",
    Hardware: "DAIZOdigitalSTROMServer",
    Revision: "00000",
    Serial: "00000000000000000000",
    EthernetID: "a8:99:5c:c0:00:27",
    MachineID: "603a932537518b121da4ffad00000037",
    Kernel: "Linuxversion2.6.32.8[jin@vsrv-pilot-feedback][gccversion4.3.3(GCC)][#1MonJan3118:55:47CET2011"
  }
}
```

9.1.2 time

Gets the installation time.

Synopsis

HTTP GET /json/system/time

Parameter

None

Response

HTTP Status 200

time	number of seconds since the Epoch, 1970-01-01 00:00:00 +0000 (UTC)
offset	offset in seconds east to GMT
daylight	boolean flag indicating if daylight saving is currently active
timezone	timezone description string

Sample

```
GET /json/system/time
{
  ok: true,
  result:
  {
    time: 1448982580,
    tzoffset: 3600,
    daylight: false,
    timezone: "Europe/Berlin"
  }
}
```

9.1.3 getDSID

Returns the dSUID and dSID of the digitalSTROM Server.

Synopsis

HTTP GET /json/system/getDSID

Parameter

None

Response

HTTP Status 200

dSID	dSID = SGTIN-96 of the dSS
dSUID	dSUID of the dSS

Sample

```
GET /json/system/getDSID
{
  ok: true,
  result :
  {
    dSID: "303505d7f800182000c00027",
    dSUID: "303505d7f8000000000182000c0002700"
  }
}
```

9.2 Authentication

9.2.1 login

Creates a new session using the provided credentials.

Synopsis

HTTP GET /json/system/login

Parameter

Parameter	Description	Remarks
user	user name string	Mandatory
password	password string	Mandatory

Response

HTTP Status 200

result.token	session token as string
--------------	-------------------------

Sample

```
GET /json/system/login?user=dssadmin&password=dssadmin
{
  "ok" : true,
  "result" : { "token" : "cea026b6f9d69e57e030736076285da77dbf117d24dbec69e349b2fb4ab7425e" }
}
```

9.2.2 logout

Destroys the session and signs out the user.

Synopsis

HTTP GET /json/system/logout

Parameter

None

Response

HTTP Status 200

Sample

```
GET /json/system/logout { "ok" : true }
```

9.2.3 loggedInUser

Returns the name of the currently logged in user.

Synopsis

HTTP GET /json/system/loggedInUser

Parameter

None

Response

HTTP Status 200

result.name	name of the currently logged in user
-------------	--------------------------------------

Note: if noone is currently logged in, the result will be empty, i.e. name will be missing.

Sample

```
GET /json/system/loggedInUser
{"ok": true, "result": {"name": "dssadmin"}}
```

9.2.4 requestApplicationToken

Returns a token for passwordless login. The token will need to be approved by a user first, the caller must not be logged in.

Synopsis

HTTP GET /json/system/requestApplicationToken

Parameter	Description	Remarks
applicationName	name of the application that requests the token	Mandatory

Parameter

Response

HTTP Status 200

result.applicationToken	application token as string
-------------------------	-----------------------------

Sample

```
GET /json/system/requestApplicationToken?applicationName=Example
{
  "ok": true,
  "result": {
    "applicationToken": "4fa07386c77d7f32260066c83b58aece5814698376bd03f0e3b5764e58f0ec1a"
  }
}
```

9.2.5 enableToken

Enables an application token, caller must be logged in.

Synopsis

HTTP GET /json/system/enableToken

Parameter

Parameter	Description	Remarks
applicationToken	application token as string	Mandatory

Response

HTTP Status 200

Sample

```
GET /json/system/enableToken?applicationToken=4fa07386c77d7f32260066c83b58aece5814698376bd03f0e3b5764e58f0ec1a
{"ok": true }
```

9.2.6 revokeToken

Revokes an application token, caller must be logged in.

Synopsis

HTTP GET /json/system/revokeToken

Parameter	Description	Remarks
applicationToken	application token as string	Mandatory

Parameter

Response

HTTP Status 200

Sample

```
GET /json/system/revokeToken?applicationToken=4fa07386c77d7f32260066c83b58aece5814698376bd03f0e3b5764e58f0ec1a
{"ok": true }
```

9.2.7 loginApplication

Creates a new session using the registered application token.

Synopsis

HTTP GET /json/system/loginApplication

Parameter	Description	Remarks
loginToken	application token as string	Mandatory

Response

HTTP Status 200

result.token	session token as string
--------------	-------------------------

Sample

```
GET /json/system/loginApplication?loginToken=4fa07386c77d7f32260066c83b58aece5814698376bd03f0e3b5764e58f0ec1a
{
  "ok" : true,
  "result" : { "token" : "a84bfd1219512078d5537a4a5cd1c78084e6c4d3f8b0ef2ae3a2c81dff638822" }
}
```

9.3 Logging

9.3.1 logChannelSeverities

Returns all log channel severities.

Synopsis

HTTP GET /json/system/logChannelSeverities/

Response

HTTP Status 200

entries	object logChannelId -> logSeverity (fatal,warning,notice,info,debug)
---------	----------------------------------------------------------------------

Sample

```
GET /json/system/logChannelSeverities/
{
  "ok" : true,
  "result" : {
    "entries" : {
      "DSS" : "info",
      "VdcToken" : "info",
      "dsLocale" : "notice",
      "System" : "debug"
    }
  }
}
```

9.3.2 logChannelSeverities/[id]

Returns the log channel severity.

Synopsis

HTTP GET /json/system/logChannelSeverities/[id]

Response

HTTP Status 200

value	logSeverity (fatal,warning,notice,info,debug)
-------	-----------------------------------------------

Sample

```
GET /json/system/logChannelSeverities/DSS
{
  "ok" : true,
```

```
"result" : {  
  "value": "info"  
}
```

9.3.3 logChannelSeverities/[id]?method=put

Sets the log channel severity.

Synopsis

HTTP GET /json/system/logChannelSeverities/[id]?method=put&value=[severity]

Parameter	Description	Remarks
method	put	Mandatory
value	log severity (fatal,warning,notice,info,debug)	Mandatory

Parameter

Response

HTTP Status 200

Sample

```
GET /json/system/logChannelSeverities/DSS?method=put&value=debug  
{ "ok" : true }
```

10 Property Tree

10.1 Basic Property Tree Operations

10.1.1 getString

Returns the string value of the property, this call will fail if the property is not of type 'string'.

Synopsis

HTTP GET /json/property/getString

Parameter

Parameter	Description	Remarks
path	path of the property	Mandatory

Response

HTTP Status 200

result.value	string value of the property
--------------	------------------------------

Sample

```
GET /json/property/getString?path=/system/version/version
```

```
{ "ok" : true, "result" : { "value" : "1.17.3" } }
```

10.1.2 setString

Sets the string value of the property, this call will fail if the property is not of type 'string'.

Synopsis

HTTP GET /json/property/setString

Parameter

Parameter	Description	Remarks
path	path of the property	Mandatory
value	string value to set	Mandatory

Response

HTTP Status 200

Sample

```
GET /json/property/setString?path=/testpath/teststring&value=testvalue
```

```
{ "ok" : true }
```


10.1.3 getInteger

Returns the integer value of the property, this call will fail if the property is not of type 'integer'.

Synopsis

HTTP GET /json/property/getInteger

Parameter

Parameter	Description	Remarks
path	path of the property	Mandatory

Response

HTTP Status 200

result.value	integer value of the property
--------------	-------------------------------

Sample

```
GET /json/property/getInteger?path=/system/uptime
```

```
{ "ok" : true, "result" : { "value" : 7539 } }
```

10.1.4 setInteger

Sets the integer value of the property, this call will fail if the property is not of type 'integer'.

Synopsis

HTTP GET /json/property/setInteger

Parameter

Parameter	Description	Remarks
path	path of the property	Mandatory
value	integer value of the property	Mandatory

Response

HTTP Status 200

Sample

```
GET /json/property/setInteger?path=/testpath/testint&value=1
```

```
{ "ok" : true }
```

10.1.5 getBoolean

Returns the boolean value of the property, this call will fail if the property is not of type 'boolean'.

Synopsis

HTTP GET /json/property/getBoolean

Parameter

Parameter	Description	Remarks
path	path of the property	Mandatory

Response

HTTP Status 200

result.value	boolean value of the property
--------------	-------------------------------

Sample

```
GET /json/property/getBoolean?path=/config/subsystems/Metering/enabled
{ "ok" : true, "result" : { "value" : true } }
```

10.1.6 setBoolean

Returns the boolean value of the property, this call will fail if the property is not of type 'boolean'.

Synopsis

HTTP GET /json/property/setBoolean

Parameter

Parameter	Description	Remarks
path	path of the property	Mandatory
value	boolean value of the property	Mandatory

Response

HTTP Status 200

Sample

```
GET /json/property/setBoolean?path=/testpath/testbool&value=true
{ "ok" : true }
```

10.1.7 getChildren

Returns an array of child nodes.

Synopsis

HTTP GET /json/property/getChildren

Parameter

Parameter	Description	Remarks
path	path of the node	Mandatory

Response

HTTP Status 200

result[]	result is an array of child nodes
----------	-----------------------------------

Sample

```
GET /json/property/getChildren?path=/system/host/interfaces/lo
{
  "ok" : true,
  "result":
  [
    { "name" : "mac", "type" : "string"},
    { "name" : "ip", "type" : "string"},
    { "name" : "netmask", "type" : "string"}
  ]
}
```

10.1.8 getType

Returns the type of the property, this can be "none", "string", "integer" or "boolean".

Synopsis

HTTP GET /json/property/getType

Parameter

Parameter	Description	Remarks
path	path of the property	Mandatory

Response

HTTP Status 200

result.type	type of the property
-------------	----------------------

Sample

```
GET /json/property/getType?path=/system/host/interfaces/lo/mac
{"ok" : true, "result" : { "type" : "string" }}
```

10.1.9 getFlags

Returns the flag values of a property.

Synopsis

HTTP GET /json/property/getFlags

Parameter

Parameter	Description	Remarks
path	path of the property	Mandatory

Response

HTTP Status 200

result.READABLE	information about the READABLE flag
result.WRITEABLE	information about the WRITEABLE flag
result.ARCHIVE	information about the ARCHIVE flag

Sample

```
GET /json/property/getFlags?path=/system/host/interfaces/lo/mac
{"ok": true, "result": {"READABLE": true, "WRITEABLE": true, "ARCHIVE": false}}
```

10.1.10 setFlag

Sets a given flag of a property.

Synopsis

HTTP GET /json/property/setFlag

Parameter

Parameter	Description	Remarks
path	path of the property	Mandatory
flag	flag identifier	Mandatory
value	boolean flag value	Mandatory

Response

HTTP Status 200

Sample

```
GET /json/property/setFlag?path=/system/host/interfaces/lo/mac&flag=WRITEABLE&value=true
{"ok": true}
```

10.1.11 remove

Removes a property node.

Synopsis

HTTP GET /json/property/remove

Parameter

Parameter	Description	Remarks
path	path of the property	Mandatory

Response

HTTP Status 200

Sample

```
GET /json/property/remove?path=/testpath
```

```
{ "ok" : true }
```

10.2 Property Query

10.2.1 query

Returns a part of the tree specified by query. All queries start from the root. The properties to be included have to be put in parentheses. A query to get all device from zone4 would look like this: '/apartment/zones/zone4/*(ZoneID,name)'. More complex combinations (see example below) are also possible.

Synopsis

HTTP GET /json/property/query

Parameter

Parameter	Description	Remarks
query	query string	Mandatory

Response

HTTP Status 200

result.value	string value of the property
--------------	------------------------------

Sample

```
GET /json/property/query?query=/apartment/zones/{*}(ZoneID,scenes)/groups/{*}(group,name)/scenes/{*}(scene,name)
```

```
{
  "ok":true,
  "result":
  {
    "zones":
    [
      {
        "ZoneID":3663,
```

```

"groups":
[
  {
    "group":1,
    "name":"yellow",
    "scenes":
    [
      {
        "scene":5,
        "name":"demo[]scene"
      }
    ]
  },
  {
    "group":2,
    "name":"gray",
    "scenes":[]
  }
]
}
]
}

```

10.2.2 query2

Differs from query(1) only in the format of the the returned json struct.

Synopsis

HTTP GET /json/Property/query2?query=/Folder1(Property1,Property2)/Folder2(Property1) HTTP GET /json/Property/query2?query=/Folder1/Folder2/Folder3(Property1) HTTP GET /json/Property/query2?query=/Fo

Folder selects the nodes to descend, Property declares which attributes we are extracting from the current node. If no properties are declared for a folder, nothing is extracted, and the node will not show up in the resulting json structure.

Parameter

Parameter	Description	Remarks
query	query string	Mandatory

Response

HTTP Status 200

result.value	string value of the property
--------------	------------------------------

Sample

```

GET '/json/property/query2?query=/apartment/zones/*[scenes]/groups/*[name]/scenes/*[name]'

{
  "ok":true,
  "result":{
    "zone0":{
      ...
    },
    "zone6268":{
      "group0":{
        "name":"broadcast"
      }
    }
  }
}

```

```

    },
    "group1":{
      "name":"yellow",
      "scene5":{
        "name":"dining"
      },
      "scene6":{
        "name":"TV"
      },
    },
    "group2":{
      "name":"gray",
      "scene6":{
        "name":"TV"
      },
      "scene17":{
        "name":"blinds 15%"
      },
    },
  },
  ...
}

```

The difference to query1 format is, that zones/groups/scenes are not returned as arrays of elements, but each element individually as a named property. This more closely matches the query format and facilitates accessing a specific element, e.g. zone6268.group1.scene6

Mind that the zones/groups folders are not part of the resulting json structure since no attributes is extracted from them. We could re-add them to the output using the wildcard (*) property match

Different from query1, we are not extracting the zoneid and scene name attribute, since that information is already contained in the element name. Neither are there any empty scene arrays. This makes the resulting json structure quite a bit smaller and easier read by a human. Potentially the json structure uses less memory is faster to generate, transfer, parse and render by a web application

11 Database

11.1 Database Query

11.1.1 query

Returns data as a result of an SQL query.

Synopsis

HTTP GET /json/database/query

Parameter

Parameter	Description	Remarks
database	name of the database	Mandatory
sql	SQL query	Mandatory

Response

HTTP Status 200

result.data	objects representing the data in the query response
-------------	-----------------------------------------------------

Sample

```
GET /json/database/query?database=dsa&sql=select * from devices limit 1;
```

```
{
  "result":
  {
    "data":
    [
      {
        "key": "3504175fe0000000000000000000000017bf6003504175fe00000000000001000000e4f00",
        "dsuid": "3504175fe0000000000000000000000017bf600",
        "dsmdsuid": "3504175fe000000000000000000000001000000e4f00",
        "deviceid": "219",
        "productid": "1224",
        "functionid": "33027",
        "version": "833",
        "zoneid": "3663",
        "configurationid": "255"
      }
    ]
  },
  "ok": true
}
```