# Sennheiser Sound Control Protocol (SSC)

## versatile command, control, and configuration for networked audio systems

## Developer‘s guide for EW-DX

EW-DX EM 2 | EW-DX EM 2 Dante | EW-DX EM 4 Dante (Firmware 3.0.x)
CHG 70N | CHG 70N-C (Firmware 3.0.x)

**Sennheiser electronic SE & Co. KG**

Am Labor 1, 30900 Wedemark, Germany, www.sennheiser.com
Publ. 03/2024

SENNHEISER

# Contents

# 1. Introduction

Modern professional audio devices are designed as building blocks for large, complex systems.

Whereas audio signal paths have converged to industry standards a long time ago, driven by practical necessities, and only recently challenged by new transport technologies like Ethernet, the professional audio markets have not evolved a similar technological convergence in the area of remote, centralised control of systems of audio equipment (the notable historical exception being MIDI, which but has a limited scope and extensibility).

In this heterogeneous environment of diverging standards proposed by individual vendors as well as open communities, there is no existing self-evident solution to be found for the needs raised by designing professional Sennheiser audio equipment.

As a consequence, communication protocols implemented in Sennheiser products have so far been designed on a single-product or product-family basis. This has worked sufficiently well, up to the point that separately developed protocols start to concur in nexus devices or applications, like:

- Wireless Systems Manager (PC-based control application for wireless transmission)
- Sennheiser Control Cockpit
- remote channel for Sennheiser microphones
- Media Control Systems (third party products, e.g., Crestron)
- A/V studio integration (third party products, e.g., Lawo)
- smartphone or tablet apps
- future centralised Sennheiser services

It has become evident that product-specific protocols fail to scale well in nexus products because of the added complexity in re-implementing the same remote control functionality from a customer point of view in a multitude of different backwards-compatible ways. It is not feasible to add more ever different technical solutions to the existing variety --- the aim must be to define a reasonably future-proof protocol suitable for existing as well as envisioned products, devices, and services.

A broad market evaluation of existing technical solutions was performed in a joint Sennheiser PRO division working group. As a result, it turns out that Open Sound Control comes closest to the specific needs for an extensible, future-proof command, control, metering, and configuration protocol for Sennheiser products.

This document describes the specific adaption of Open Sound Control to Sennheiser use, "Sennheiser Sound Control", SSC. The main other ingredient is JavaScript Object Notation (JSON), which enhances ease-of-use and the implementation complexity for small to smallest devices.

Note that the protocol is intended for command and control. Network audio streaming is entirely out of its scope.

# 2. Open Sound Control Overview

Open Sound Control (OSC) is a protocol developed at The Center For New Music and Audio Technology (CNMAT) at University of California, Berkeley.

The OSC specification Version 1.1 is available from the Open Sound Control website at:

http://www.opensoundcontrol.org/ .

The OSC Schema defined by MicroOSC at:

http://cnmat.berkeley.edu/library/uosc_project_documentation/osc_address_schema

was used as a starting point for some parts of the schema defined in this document.

OSC handles more advanced packet formats such as bundles of messages to be atomically executed at the same time with timestamps, as well as addresses with wildcards and array values.

## JavaScript Object Notation Overview

JavaScript Object Notation (JSON) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Ruby, Python, and many others. These properties make JSON an ideal data-interchange language.

The central website for JSON information is http://json.org. JSON is formally specified in RFC 4627 (MIME-type application/json).

JavaScript Object Notation (JSON) is a text format for the serialization of structured data. It is derived from the object literals of JavaScript, as defined in the ECMAScript Programming Language Standard, Third Edition.

JSON can represent four primitive types (strings, numbers, booleans, and null) and two structured types (objects and arrays).

A string is a sequence of zero or more Unicode characters.

An object is an unordered collection of zero or more name/value pairs, where a name is a string and a value is a string, number, boolean, null, object, or array.

An array is an ordered sequence of zero or more values.

The terms "object" and "array" come from the conventions of JavaScript.

JSON’s design goals were for it to be minimal, portable, textual, and a subset of JavaScript.

# 3. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP14/RFC 2119, "Key words for use in RFCs to Indicate Requirement Levels".

## Terminology

| | |
|---|---|
| SSC Message | protocol unit of transmission |
| SSC Server | device, application or person that sends SSC messages |
| SSC Container | named entity containing SSC Methods or other Containers |
| SSC Method | named attribute or action callable on a SSC Server |
| SSC Address | full name of a SSC Method, including names of all enclosing Containers. may be represented by a JSON object hierarchy. |
| SSC Address Tree | a JSON object hierarchy consisting of one or more SSC Addresses. |
| SSC Address Space | hierarchical tree comprising all the SSC Addresses of a SSC Server |
| SSC Method Call | SSC Message requesting execution of a SSC Method |
| SSC Method Arguments | arguments included in a SSC Method Call |
| SSC Method Reply | SSC Message send by SSC Server as result of a Method Call |
| binary OSC | the binary OSC encoding as opposed to JSON-based SSC |
| restricted SSC Server | a SSC Server that doesn‘t implement some optional parts of this specification |

# 4. SSC Data Structure Specification

## Applying JSON to the OSC device model

OSC models the controlled device as a tree-shaped hierarchy of methods, with the method addresses constructed from the names of all the nodes in the hierarchy, written like a file path.

| | | | |
|---|---|---|---|
| `/` | | | container at address "/" |
| | `audio/` | | container at address "/audio/" |
| | | `out1/` | container at address "/audio/out1/" |
| | | | `label name` address "/audio/out1/label": method with string argument |
| | | | `level_db 5` address "/audio/out1/level_db": method with numeric argument |
| | | | `...` more methods of "/audio/out1" |
| | | `out2/` | container at address "/audio/out2 /" |
| | | `...` | methods of "/audio/out2" |
| | `device/` | | container at address "/device/" |
| | | `...` | methods of "/device" |
| | `...` | | more methods and containers of "/" |

JSON allows to model that structure as a hierarchy of JSON objects.

| | | | |
|---|---|---|---|
| `{` | | | root object |
| `"audio": {` | | | object "audio" |
| | `"out1": {` | | object "audio.out1" |
| | | `"label": name,` | numerical property "audio.out1.label" |
| | | `"level_db": 5,` | boolean property "audio.out1.level_db" |
| | | `...` | more properties of "audio.out1" |
| | `},` | | |
| | `"out2": {` | | object "audio.out2" |
| | | `...` | properties of "audio.out2" |
| | `},` | | |
| `"device": {` | | | object "device" |
| | `...` | | properties of "device" |
| `},` | | | |

| | |
|---|---|
| `...` | more properties and objects of the root object |

`}`

The OSC Method Address (like "/audio/out1/level_db") is interpreted as a property path navigating through the hierarchy of JSON objects. The value of each property MUST be either a primitive JSON data type, or a JSON array. Rationale: This allows to clearly separate SSC Method Addresses from SSC Method Arguments at JSON parser level without knowledge of the underlying method address tree.

The resulting JSON tree structure of hierarchical objects, the SSC Address Space, is tailored to describe the functionality of a specific SSC Server, in the same way as foreseen by OSC.

In JSON it is possible to serialise the complete state of all properties in the tree to a closed form, thus describing the complete state of the SSC Server. In this way, JSON can be used as an excellent extensible data format for configuration files, or for scripting applications, which drive a system of SSC Servers through a sequence of programmed configurations.

For command and control applications it is desirable to access single properties independently. This can be achieved in JSON syntax by the simple convention, that all the properties of a SSC Server that are not mentioned in a JSON message are left unchanged.

In this way, applied to the example above, the JSON form

```
{ "audio": { "out1": { "level_db": 5 } } }
```

can be understood as a SSC Method Call of the SSC Method "/audio/out1/level_db" with the argument 5, presumably to set the level to that level, or as an SSC Method Reply message stating the current level.

# JSON Message Transaction Syntax

The SSC Message exchange is described here as transaction using the following syntax:

Prefix "TX:" indicates a SSC Message that a SSC Client is sending to a SSC Server.

Prefix "RX:" indicates a SSC Message that the SSC Server will send back to the Client.

A SSC-Message is written verbatim, enclosed by curly brackets `{ }`.

A transaction to set the level of "audio" of "out1" to 1 then looks like this:

```
TX: { "audio": { "out1": { "level_db": 1 }}}
```

```
RX: { "audio": { "out1": { "level_db": 1 }}}
```

Note that the execution of the method results in a method reply message, which for simple property setters states the actual value of the property resulting from executing the message.

The resulting value may be different from the supplied argument, e.g., for a read-only property, or if the argument is out of range, and the device may adapt it to the allowed range (this is not considered as an error):

```
TX: { "audio": { "out1": { "level_db": 20000 }}}
```

```
RX: { "audio": { "out1": { "level_db": 6 }}}
```

Getter-methods, which request the value of a property from the SSC Server, are realised by supplying the special JSON value null as argument to method sent to the address of the property:

```
TX: { "audio": { "out1": { "level_db": null }}}
```

```
RX: { "audio": { "out1": { "level_db": 6 }}}
```

Compared to binary OSC, the JSON syntax is slightly more verbose for single attribute settings, but this is compensated when multiple attributes are set in the same transaction:

```
TX: { "audio": { "out1": { "level_db": 3, "label": null }}}
```

```
RX: { "audio": { "out1": { "level_db": 3, "label": "AF_LABEL" }}}
```

# SSC JSON Message Syntax

## Elementary data types

All SSC data is composed of the primitive JSON data types:

- string: a sequence of zero or more Unicode characters in UTF-8 encoding, wrapped in double quotes, using backslash escapes. A character is represented as a single character string. Binary zero bytes can be included in a string using Unicode escape notation: "\u0000".
- number: a number in conventional "scientific" notation. `0, 42, -23, 3.141259, 1.0e+100` are all valid numbers. A Restricted SSC Server MAY reject non-integer numeric arguments, or it MAY adapt them by silently converting them to integer values.
- `true`: the boolean true value.
- `false`: the boolean false value.
- `null`: indicates a missing value; used as pseudo argument for getter-methods.

## SSC Messages

A Message is the protocol unit of transmission. Any application that sends SSC Messages is a SSC Client, any application that receives SSC Messages is a SSC Server.

A SSC Message MUST be sent as a single closed JSON form describing a JSON object. Extra whitespace between the elements of the message MUST be ignored by the receiver.

This means that every SSC Message is enclosed in a pair of curly brackets `{ }`.

## SSC Addresses

Every SSC Server implements a set of SSC Methods. SSC Methods are the potential destinations of SSC Messages received by the SSC Server, and correspond to each of the points of control that the application makes available. "Invoking" a SSC Method is analogous to a procedure call; it means supplying the method with arguments and causing the method's effect to take place. The SSC Server MUST respond to each received SSC Message by sending a SSC Method Reply Message to the originating SSC Client.

A SSC Server's SSC Methods are arranged in a tree structure called a SSC Address Space. The leaves of this tree are the SSC Methods and the branch nodes are called SSC Containers. A SSC Server's SSC Address Space MAY be dynamic; that is, its contents and shape MAY change over time.

Each SSC Method and each SSC Container other than the root of the tree MUST have a symbolic name which MUST be composed entirely of printable ASCII characters other than the following:

| | | |
|---|---|---|
| " " | space, | ASCII 32 |
| " | double quote, | ASCII 34 |
| # | number sign, | ASCII 35 |
| * | asterisk, | ASCII 42 |
| , | comma, | ASCII 44 |
| / | slash, | ASCII 47 |
| : | colon, | ASCII 58 |
| ? | question mark, | ASCII 63 |
| [ | open bracket, | ASCII 91 |
| ] | close bracket, | ASCII 93 |
| { | open curly brace, | ASCII 123 |
| } | close curly brace, | ASCII 125 |

The SSC Address of a SSC Method is a symbolic name giving the full path to the SSC Method in the SSC Address Space, starting from the root of the tree. A SSC Method's SSC Address begins with the character "/" (forward slash), followed by the names of all the containers, in order, along the path from the root of the tree to the SSC Method, separated by forward slash characters, followed by the name of the SSC Method. The syntax of SSC Addresses was chosen to match the syntax of URLs. The SSC address syntax SHOULD be used in documentation, but it SHOULD NOT be used as an argument to other SSC Methods; the JSON syntax of hierarchical objects SHOULD be used instead.

A SSC Method may be invoked with an empty argument list by supplying the JSON null value. This kind of SSC Method call SHOULD normally have the semantics of a query resulting in the current value of the property addressed by the method, without further side effects. SSC Methods that change the state of a SSC Server SHOULD normally have arguments.

Example:

- query current level of OUT1 output of AUDIO module:
  - `TX: { "audio": { "out1": { "level_db": null }}}`
  - `RX: { "audio": { "out1": { "level_db": 5 }}}`
- change level of OUT1 output of AUDIO module (note that the server adapts the value):
  - `TX: { "audio": { "out1": { "level_db": 999 }}}`
  - `RX: { "audio": { "out1": { "level_db": 6 }}}`

# 5. SSC subscriptions - /osc/state/subscribe

A subscription request is sent by a client to a server for an address pattern to subscribe to. The SSC Server normally accepts the subscription request, and remembers that the requesting client wishes to be notified about value changes of the subscribed addresses.

The SSC Server MAY refuse subscription requests, subject to device-specific policy or implementation specific limitations. The SSC Server MUST reply on the subscription request immediately either by acknowledging the request, or by sending an error reply.

The SSC Server MUST send an initial subscription notification to the client, which contains the result of calling the subscribed SSC Methods immediately with null-argument when the subscription request is handled. This initial notification MAY be bundled with the reply to the subscription request itself.

Each subscription notification MUST have identical contents to the reply to an imagined SSC Method invocation with null-argument to the subscribed SSC Method Address at the time that the notification is sent.

The SSC Client MAY bundle a call to /osc/xid with the subscription request. If a xid is supplied, a reply to /osc/xid MAY be bundled with each subscription notification, with the xid of the reply identical to that supplied by the client.

The SSC Server MUST send value changes of the subscribed addresses to the SSC Client. By default, the SSC Server will send subscription notifications if and only if the subscribed addresses change in value. The SSC Client can modify this behaviour by supplying optional parameters with the subscription request, allowing to either throttle the rate of notifications, or stimulate additional periodic notifications even if the subscribed addresses do not change in value.

Every subscription is specific to the connection between SSC Client and SSC Server. Also each SSC Method can only be subscribed once per connection. This means, that if a SSC Client requests a subscription which is already subscribed by that client on that connection, then the SSC Server MUST treat this as if the existing subscription was silently terminated and immediately requested anew.

## Subscription notification rate parameters

Optional subscription request parameters related to notification rate:

- "min" minimum notification period (ms), 0=none, default 0
- "max" maximum notification period (ms), 0=none, default 0

If "min" is 0, then notifications are not sent when a subscribed address changes in value, they are only sent based on the "max" period. If "min" is greater than 0, notifications are sent after the specified time duration has elapsed, even if the value of the subscribed address is unchanged. If "max" is 0, then notifications are only sent when a value changes, or based

on the "min" period. If "max" is greater than 0, then notifications are sent not earlier before the specified time duration has elapsed, even if the subscribed address changes value in the meantime.

## Subscription cancelling and expiration

The SSC Server MUST terminate a subscription in these cases:

- the subscribed client cancels the subscription explicitly
- a maximum number of notifications has been sent
- a maximum lifetime relating to the begin of the subscription expires
- the SSC Client closes the connection
- the transport layer of the SSC connection signals a communication error

If the SSC Server decides to terminate the connection because the lifetime or notification count expires, then it MUST inform the SSC Client by sending an error reply "310 – subscription terminated" to the SSC address that terminates subscription together with or immediately after the last subscription notification.

Optional subscription request parameters related to termination:

| | |
|---|---|
| "cancel" | "true" cancels the subscription (default false). |
| "count" | maximum number of notifications to send, default 1000 |
| "lifetime" | maximum lifetime (s) of the subscription, default 10s |

The SSC Client may renew a subscription at any time, thereby resetting all of the lifetime limitations. To renew a subscription, the SSC Client re-requests it; there’s no difference between an initial subscription request and a renewal request.

## Subscribing to multiple addresses

The SSC Client MAY request multiple subscriptions in a single request; either by providing them explicitly as SSC Address Tree, or by specifying address patterns as subscription addresses, or even both in the same request.

The SSC Server MAY either treat all those subscription requests separately, as if the addresses had all been requested for subscription individually. In this case all the subscription notifications would each contain the SSC Method Reply to a single subscribed address.

Alternatively, the SSC Server MAY bundle subscription notifications which happen to be sent at the same time into a single notification. The SSC Client MUST be able to handle a bundled notification if it requests multiple subscriptions in a single request, but it MUST NOT rely on the SSC Server bundling the notifications.

In any case the SSC Server SHOULD NOT bundle notification causes, meaning that the SSC Server SHOULD NOT send any subscription notifications for addresses in a bundle with notifications to other addresses, if they would not be sent if all subscriptions had been requested individually.

If some of the SSC addresses in a subscription request must be rejected with errors, whereas other subscriptions succeed, then the SSC Server MAY reject the request completely with an error reply detailing all the failed addresses. If possible, the SSC Server SHOULD instead execute the successful subscriptions and only reject the erroneous ones. This MUST result in a successful reply message to the subscription request, with the reply value including only the successful addresses. In this case the SSC Error state MUST be set to "210 – Partial Success", and MAY be accompanied by a parameter named "failed_addresses" with an Array of Address trees composed of all the failed Method Addresses (erroneous Addresses replaced by {}), in bundled or unbundled representation. The value of the Address in the Address Tree SHOULD be set to the SSC Error Code relating to the failure of the specific Address. See also the transaction example.

The SSC Server MAY also send a SSC Error "210 – Partial Success" when in fact all of the subscriptions have failed, because the SSC Client receives sufficient information in this Error Reply to work out this fact.

# Supscription request and reply syntax

The SSC Address for subscriptions is `/osc/state/subscribe`.

This SSC Method may be called with a null parameter, which results in a SSC Address tree of all addresses currently subscribed by the SSC Client on the current connection.

The SSC Method also takes a structured parameter, specified as a JSON array.

Each element of the array is a SSC Address Tree specifying the SSC addresses that the SSC Client requests to subscribe. The SSC Address Tree MAY contain Address patterns.

A SSC Server that supports subscription MUST be able to interpret a single Address Tree element in the Method Argument array. Multiple Address Trees MAY be supported, or the SSC Server MAY reject them with a SSC Error 414 (request too complex).

The Response to the subscription Request will normally echo the Request, if all subscriptions can be handled successfully. If subscription parameters were requested, then the SSC Server MAY adapt the requested parameters, and MUST send back the adapted parameter values in the Reply. If multiple subscriptions are requested in a single Request, then the SSC Server might find it necessary to adapt subscription parameters differently for different Addresses. In that case, the array in the Reply MAY contain additonal Address trees containing additional adapted parameter objects. The SSC Server MAY also reject the subscription request completely (with SSC Error code 406), or partially (with SSC Error code 210) in such a case.

# 6. SSC Transport Layer Adaptations

The SSC data format as defined in the previous sections can be transported by different transport protocols, or stored in persistent files. This section specifies what transports are supported, and how the specific features of transport layers shall be applied to transporting SSC Messages.

## UDP/IP

UDP/IP is the standard transport for all devices with an Ethernet interface or another interface typically used for internet connectivity. All those device MUST implement the UDP/IP transport for SSC.

The device implements UDP over IPv4.

One UDP datagram is used to transport one SSC Message. If the SSC Message is really large (e.g., a complete device configuration), IP fragmentation might fail, if a restricted device does not implement IP re-assembly properly. In that case, the SSC Server should break up the message into multiple SSC Method Calls instead. If atomic execution is relevant, SSC time tags may be used.

The UDP port number to used by the SSC Server should normally be discovered by the SSC Client by means of the server discovery protocol. The default port number is 45.

Rationale: No other standard UDP service is expected to use 45. The IANA reservation for a "Message Passing Service" is historic, and SSC is actually passing messages itself. Sennheiser was founded in 1945.

# SSC Server Discovery

Networked devices implement DNS-SD (Apple Bonjour) as discovery protocol.

The DNS Service-Type is specified as "_ssc".

Because all networked SSC Servers must implement SSC-over-UDP, they MUST all publish a DNS-SD service under "_ssc. _udp". Those servers that additionally support TCP MUST publish another DNS-SD service under "_ssc._tcp".

The DNS-SD service instance name must be identical to the device name accessible as /device/name. DNS-SD automatic name collision resolution SHOULD be performed, and the resulting name changes MUST be reflected back into /device/name and the persistent device configuration. The renaming rules MAY be tailored to suit product specific requirements.

The DNS-SD service registration includes the port numbers used. SSC Clients SHOULD NOT rely on default ports.

The DNS-SD hostname SHOULD NOT be presented to the user. It may contain a unique identification part (e.g., derived from the device MAC or serial), to avoid name collisions and automatic renaming.

Additional information about the SSC Server may be provided with a DNS-SD TXT-record.

The following properties are currently defined for the TXT record:

| | |
|---|---|
| txtvers | Version of the TXT record format. Currently "1". |
| version | SSC-Version provided by the SSC Server. |

# 7. Developer’s Guide for EW-DX EM

This chapter describes in detail how a developer should use the SSC interface as implemented for the EW-DX EM 2.

## Limitations

### SSC Transport Layer

The SSC Server implemented for EW-DX devices supports only UDP/IP as transport protocol. All the devices support IPv4.

### Subscriptions

The EW-DX EM 2 receiver supports SSC Method subscriptions from up to eight different SSC clients simultaneously.

# 8. SSC Method List (EW-DX EM)

## "/device/identification/visual"

Command to start/stop device identification visual (true: start; false: stop; null: read status).

- type: Read/Write
- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/device/identity/version"

Product version.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false

# "/device/identity/vendor"

Command replies "Sennheiser electronic SE & CO KG".

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false

# "/device/identity/serial"

Not yet implemented. Serial no from production.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false

# "/device/identity/product"

Product label

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false

# "/device/network/dante/identity/version"

EW-DX EM 2 Dante and EW-DX EM 4 Dante exclusive! Get version of DANTE SW as string value.

- type: Read Only
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/device/network/dante/ipv4/netmask"

EW-DX EM 2 Dante and EW-DX EM 4 Dante exclusive! Get the current network IPv4 netmask addresses of Dante interfaces `[primary, secondary]` as array of string values e.g. "255.255.255.0","0.0.0.0".

- type: Read Only
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 2
    - subscr: true

# "/device/network/dante/ipv4/manual_netmask"

EW-DX EM 2 Dante and EW-DX EM 4 Dante exclusive! Set/get manual network IPv4 netmask address of Dante interfaces `[primary, secondary]` as array of string values e.g. "255.255.255.0","255.255.255.0".

- type: Read/Write
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - count: 2
    - subscr: true

# "/device/network/dante/ipv4/manual_ipaddr"

EW-DX EM 2 Dante and EW-DX EM 4 Dante exclusive! Set/get manual network IPv4 IP address of Dante interfaces `[primary, secondary]` as array of string values e.g. "192.168.1.10","192.168.2.10".

- type: Read/Write
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - count: 2
    - subscr: true

# "/device/network/dante/ipv4/manual_gateway"

EW-DX EM 2 Dante and EW-DX EM 4 Dante exclusive! Set/get manual network IPv4 gateway address of Dante interfaces `[primary, secondary]` as array of string values e.g. "192.168.1.1","192.168.2.1".

- type: Read/Write
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - count: 2
    - subscr: true

# "/device/network/dante/ipv4/ipaddr"

EW-DX EM 2 Dante and EW-DX EM 4 Dante exclusive! Get the current network IPv4 IP addresses of Dante interfaces `[primary, secondary]` as array of string values e.g. "192.168.1.10","0.0.0.0".

- type: Read Only
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 2
    - subscr: true

# "/device/network/dante/ipv4/gateway"

EW-DX EM 2 Dante and EW-DX EM 4 Dante exclusive! Get the current network IPv4 gateway addresses of Dante interfaces `[primary, secondary]` as array of string values e.g. "192.168.1.1","0.0.0.0".

- type: Read Only
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 2
    - subscr: true

# "/device/network/dante/ipv4/auto"

EW-DX EM 2 Dante and EW-DX EM 4 Dante exclusive! Set/read network mode of Dante interfaces `[primary, secondary]` -> true: Auto-IP/DHCP; false: Manual.

- type: Read/Write
- value: Boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - count: 2
    - subscr: true

# "/device/network/dante/macs"

EW-DX EM 2 Dante and EW-DX EM 4 Dante exclusive! Get DANTE MAC addresses as array of string values.

- type: Read/Write
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - subscr: true

# "/device/network/dante/interfaces"

EW-DX EM 2 Dante and EW-DX EM 4 Dante exclusive! Get DANTE interfaces names as array of string values.

- type: Read Only
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1

# "/device/network/dante/interface_mapping"

EW-DX EM 2 Dante and EW-DX EM 4 Dante exclusive! Set/get the dante interface or port mapping configuration ID as string.

- type: Read/Write
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ SINGLE_CABLE: single,off,single,single,off,single
        - ▷ SPLIT1: ctrl,off,ctrl,pri,off,ctrl
        - ▷ SPLIT2: ctrl,off,pri,pri,off,ctrl
        - ▷ SPLIT: ctrl,off,pri,pri,off,ctrl
        - ▷ AUDIO_REDUNDANCY: ctrl,off,sec,pri,off,ctrl
    - subscr: true

# "/device/network/ether/macs"

List of MAC addresses.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/device/network/ether/interfaces"

List of all ethernet interfaces.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/device/network/ipv4/manual_netmask"

Static (Manual) Netmask like "255.255.255.0".

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - count: 1
    - subscr: true

# "/device/network/ipv4/manual_ipaddr"

Static (Manual) IPv4 addr like "192.168.1.203".

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - count: 1
    - subscr: true

# "/device/network/ipv4/manual_gateway"

Static (Manual) Gateway addr like "192.168.1.1".

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - count: 1
    - subscr: true

# "/device/network/ipv4/netmask"

Online Netmask like "255.255.255.0".

- type: Read-only
- value: String
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 1
    - subscr: true

# "/device/network/ipv4/ipaddr"

Online IPv4 addr like "192.168.1.203".

- type: Read-only
- value: String
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 1
    - subscr: true

# "/device/network/ipv4/interfaces"

List of ipv4 interface indices.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/device/network/ipv4/gateway"

Online Gateway addr like "192.168.1.1".

- type: Read-only
- value: String
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 1
    - subscr: true

# "/device/network/ipv4/auto"

Command to configure ip settings automatically (true: use DHCP and ZeroConf (Auto-IP); false: set ip address/netmask/gateway manually).

- type: Read/Write
- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/device/network/mdns"

Command to enable, disable or read mdns state.

- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - count: 1
    - subscr: true

# "/device/preset_spacing"

get device preset spacing in kHz depending on if LD mode is active or not.

- type: Read/Write
- value: Number.
- example: `{"device":{"preset_spacing":null}} -> reply: {"device":{"preset_spacing":300}}`
- limits
    - type: Number
    - const: true
    - writeable: false
    - units: kHz

# "/device/time"

Current absolute clock value (in seconds) of the device since 2000-01-01T00:00:00+0000 UTC.

- type: Read * value: Integer
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: s
    - max: -1
    - min: 0
    - inc: 1

# "/device/restore"

Command to restore device default settings. Valid options: "FACTORY_DEFAULTS" (restore all factory defaults), "AUDIO_DEFAULTS" (restore audio defaults only).

- type: write-only
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ FACTORY_DEFAULTS = Restore all factory defaults
        - ▷ AUDIO_DEFAULTS = Restore audio defaults only

# "/device/restart"

Restarts the device

- limits
    - type: Boolean
    - const: false
    - writeable: true

# "/device/remote_access"

You can store a 15 characters long string here. It will not be stored persistently and will be cleared after 2 minutes!

- type: Read/Write
- value: string
- example: `{"sys":{"remote":"192.168.111.222"}}`
- limits
    - type: String
    - const: false
    - writeable: true
    - subscr: true

# "/device/name"

Command to set or read device name.

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - length: 18
    - subscr: true

# "/device/lock"

Set EW-DX EM device keys locked (true) / unlocked (false)

- type: Read/Write
- value: bool
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/device/location"

Command to set or read device location.

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - length: 400
    - subscr: true

# "/device/link_density_mode"

Switch RF transmission link density mode on if true is specified or off (standard mode) if false is specified. Note: This command leads to a device reboot!

- type: Read/Write
- value: Boolean
- example: `{"device":{"link_density_mode":true}}`
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/device/language"

List of supported languages. English only = `["en_GB"]`.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/device/frequency_ranges"

Command to read receiver device frequency ranges. Format of replied string array: `["<range_1 start frequency>:<range_1 step>:<range_1 end frequency>, <range_2 start frequency>:<range_2 step>:<range_2 end frequency>,..., <range_n start frequency>:<range_n step>:<range_n end frequency>"]`

- type: Read-only
- value: String
- example: `{"device":{"frequency_ranges":["470000000:25000:514875000","522100000:25000:550000000"]}}` means `[470MHz..514.875MHz]`, `[522.1MHz..550MHz]` step=25kHz
- limits
    - type: String
    - const: true
    - writeable: false
    - count: -1

# "/device/frequency_code"

Command to read receiver device frequency range code.

- type: Read-only
- value: String
- example: `{"device":{"frequency_code":"Q1-9"}}`
- limits
    - type: String
    - const: true
    - writeable: false

# "/device/encryption"

Activate/Deactivate encryption on the device.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/device/brightness"

Brightness in percent index (1..5) corrresponds to (20%..100%).

- type: Read/Write
- value: number
- example: `{"device":{"brightness":5}}`
- limits
    - type: Number
    - const: false
    - writeable: true
    - max: 5
    - min: 1
    - inc: 1
    - options , option_desc
        - ▷ 1 = 20%
        - ▷ 2 = 40%
        - ▷ 3 = 60%
        - ▷ 4 = 80%
        - ▷ 5 = 100%
    - subscr: true

# "/device/booster"

EW-DX EM 4 Dante exclusive! Supply power for a external antenna booster. Write values: `[true, false]` -> `[On, Off]`

- type: Read/Write
- value: Boolean
- example: `{"device":{"booster":true/false/null}}`
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/interface/version"

Command to read Win-Plus SSC interface version.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false

# "/osc/state/auth/access"

Read authentication access of current SSC client. Response value is an array of strings, comma separated

- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - subscr: true

# "/osc/state/prettyprint"

SSC reply output style is not supported. Returns false.

- limits (hidden)

# "/osc/state/close"

SSC connection close.

- limits (hidden)

# "/osc/state/subscribe"

SSC subscriptions.

- limits (hidden)

# "/osc/feature/timetag"

SSC timed method execution is not supported. Returns false.

- limits (hidden)

# "/osc/feature/baseaddr"

SSC interactive method address base is not supported. Returns false.

- limits (hidden)

# "/osc/feature/subscription"

SSC subscriptions are supported. Returns true.

- limits (hidden)

# "/osc/feature/pattern"

SSC message dispatching and pattern matching are supported. Returns "*?".

- limits (hidden)

# "/osc/limits"

SSC method parameter range reflection.

- limits (hidden)

# "/osc/schema"

SSC schema reflection.

- limits (hidden)

# "/osc/version"

SSC protocol version.

- limits (hidden)

# "/osc/xid"

SSC transaction ID.

- limits (hidden)

# "/osc/ping"

SSC Ping.

- limits (hidden)

# "/osc/error"

SSC error state.

- limits (hidden)

# "/rx1/identification/visual"

Command to start/stop receiver channel 1 identification visual (true: start; false: stop; null: read status).

- type: Read/Write
- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/presets/user/0"

set/get list of user frequencies `[kHz]` for receiver channel 1.

- type: Read/Write
- value: Number.
- example: `{"rx1":{"presets":{"user":{"0":[549200, 549800]}}}}`
- limits
    - type: Number
    - const: false
    - writeable: true
    - count: -1
    - subscr: true

# "/rx1/presets/active"

set/get frequency from specific (presets) frequency list for receiver channel 1.

- type: Read/Write
- value: Number.
- example: select from factory list: index=0, channel=2: `{"rx1":{"presets":{"active":"factory:0:2"}}}` or select from user list: index=0, channel=1: `{"rx1":{"presets":{"active":"user:0:1"}}}`
- limits
    - type: String
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/sync_settings/trim_ignore"

Skip/apply trim setting on TX sync action for receiver channel 1.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/sync_settings/trim"

TX trim for receivder channel 1.

- type: Read/Write
- value: Number
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 6
    - min: -12
    - inc: 1
    - subscr: true

# "/rx1/sync_settings/name_ignore"

Skip/apply channel name setting on TX sync action for receiver channel 1.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/sync_settings/mute_config_ts"

TS mute configuration (mute mode) for receiver channel 1.

- type: Read/Write
- value: string
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ af_mute = AF Mute
        - ▷ off = Disabled
        - ▷ push_to_talk = Push to talk
        - ▷ push_to_mute = Push to mute
    - subscr: true

# "/rx1/sync_settings/mute_config_ignore"

Skip/apply mute configuration (mute mode) setting on TX sync action for receiver channel 1.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/sync_settings/mute_config"

TX mute configuration (mute mode) for receiver channel 1.

- type: Read/Write
- value: string
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off = Disabled
        - ▷ rf_mute = RF Mute
        - ▷ af_mute = AF Mute
    - subscr: true

# "/rx1/sync_settings/lowcut_ignore"

Skip/apply low cut frequency setting on TX sync action for receiver channel 1.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/sync_settings/lowcut"

TX low cut frequency for receiver channel 1.

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off
        - ▷ 30 Hz
        - ▷ 60 Hz
        - ▷ 80 Hz
        - ▷ 100 Hz
        - ▷ 120 Hz
    - subscr: true

# "/rx1/sync_settings/lock_ignore"

Skip/apply Auto Lock setting on TX sync action for receiver channel 1.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/sync_settings/lock"

TX Autolock off/on for receiver channel 1.

- type: Read/Write
- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/sync_settings/led_ignore"

Skip/apply LED setting on TX sync action for receiver channel 1.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/sync_settings/led"

TX LED off/on for receiver channel 1.

- type: Read/Write
- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/sync_settings/frequency_ignore"

Skip/apply frequency setting on TX sync action for receiver channel 1.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/sync_settings/cable_emulation_ignore"

Skip/apply cable emulation setting on TX sync action for receiver channel 1.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/sync_settings/cable_emulation"

TX cable emulation for receiver channel 1.

- type: Read/Write
- value: string
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off
        - ▷ type1
        - ▷ type2
        - ▷ type3
    - subscr: true

# "/rx1/warnings"

Active warnings on receiver channel 1 stored in an array of strings. Warnings: `[Aes256Error, LowSignal, NoLink, RfPeak, AfPeak]`.

- type: Read only
- value: String
- example: `{"rx1":{"warnings":null}} -> {"rx1":{"warnings":["Aes256Error","AfPeak"]}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - options:
        - ▷ Aes256Error
        - ▷ LowSignal
        - ▷ NoLink
        - ▷ RfPeak
        - ▷ AfPeak
    - subscr: true

# "/rx1/name"

User-settable channel name for receiver channel 1.

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - length: 8
    - subscr: true

# "/rx1/mute"

Set audio output mute on/off of receiver channel 1.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx1/mates"

Information about active mates of the receiver channel 1 stored as array of string values.

- type: Read only
- value: String
- example: `{"rx1":{"mates":null}} -> {"rx1":{"mates":["mates/tx1"]}}`
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/rx1/gain"

Gain level for receiver channel 1. Valid values in range `[-3dB..+42dB]`, increment step = 3dB.

- type: Read/Write
- value: Number
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 42
    - min: -3
    - inc: 3
    - subscr: true

# "/rx1/frequency"

Carrier Frequency in kHz for receiver channel 1.

- type: Read/Write
- value: Number.
- example: `{"rx1":{"frequency":null/520000}}`
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: kHz
    - max: 1999000
    - min: 470200
    - inc: 25
    - subscr: true

# "/rx1/audio"

Information about active audio inputs/outputs of the receiver channel 1 stored as array of string values.

- type: Read only
- value: String
- example: `{"rx1":{"audio":null}} -> {"rx1":{"audio":["audio1/out1"]}}`
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/rx1/restore"

Command to restore device audio default settings of receiver channel 1. Valid options: "AUDIO_DEFAULTS" (restore audio defaults only).

- type: write-only
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ AUDIO_DEFAULTS = Restore audio defaults only

# "/rx2/identification/visual"

Command to start/stop receiver channel 2 identification visual (true: start; false: stop; null: read status).

- type: Read/Write
- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/presets/user/0"

set/get list of user frequencies `[kHz]` for receiver channel 2.

- type: Read/Write
- value: Number.
- example: `{"rx2":{"presets":{"user":{"0":[548000, 5483000, 548600]}}}}`
- limits
    - type: Number
    - const: false
    - writeable: true
    - count: -1
    - subscr: true

# "/rx2/presets/active"

set/get frequency from specific (presets) frequency list for receiver channel 2.

- type: Read/Write
- value: Number.
- example: select from factory list: index=0, channel=2: `{"rx2":{"presets":{"active":"factory:0:2"}}}` or select from user list: index=0, channel=1: `{"rx2":{"presets":{"active":"user:0:1"}}}`
- limits
    - type: String
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/sync_settings/trim_ignore"

Skip/apply trim setting on TX sync action for receiver channel 2.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/sync_settings/trim"

TX trim for receivder channel 2.

- type: Read/Write
- value: Number
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 6
    - min: -12
    - inc: 1
    - subscr: true

# "/rx2/sync_settings/name_ignore"

Skip/apply name setting on TX sync action for receiver channel 2.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/sync_settings/mute_config_ts"

TS mute configuration (mute mode) for receiver channel 2.

- type: Read/Write
- value: string
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ af_mute = AF Mute
        - ▷ off = Disabled
        - ▷ push_to_talk = Push to talk
        - ▷ push_to_mute = Push to mute
    - subscr: true

# "/rx2/sync_settings/mute_config_ignore"

Skip/apply mute configuration (mute mode) setting on TX sync action for receiver channel 2.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/sync_settings/mute_config"

TX mute configuration (mute mode) for receiver channel 2.

- type: Read/Write
- value: string
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off = Disabled
        - ▷ rf_mute = RF Mute
        - ▷ af_mute = AF Mute
    - subscr: true

# "/rx2/sync_settings/lowcut_ignore"

Skip/apply low cut frequency setting on TX sync action for receiver channel 2.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/sync_settings/lowcut"

TX low cut frequency for receiver channel 2.

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off
        - ▷ 30 Hz
        - ▷ 60 Hz
        - ▷ 80 Hz
        - ▷ 100 Hz
        - ▷ 120 Hz
    - subscr: true

# "/rx2/sync_settings/lock_ignore"

Skip/apply Auto Lock setting on TX sync action for receiver channel 2.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/sync_settings/lock"

TX Autolock off/on for receiver channel 2.

- type: Read/Write
- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/sync_settings/led_ignore"

Skip/apply LED setting on TX sync action for receiver channel 2.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/sync_settings/led"

TX LED off/on for receiver channel 2.

- type: Read/Write
- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/sync_settings/frequency_ignore"

Skip/apply frequency setting on TX sync action for receiver channel 2.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/sync_settings/cable_emulation_ignore"

Skip/apply cable emulation setting on TX sync action for receiver channel 2.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/sync_settings/cable_emulation"

TX cable emulation for receiver channel 2.

- type: Read/Write
- value: string
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off
        - ▷ type1
        - ▷ type2
        - ▷ type3
    - subscr: true

# "/rx2/warnings"

Active warnings on receiver channel 2 stored in an array of strings. Warnings: `[Aes256Error, LowSignal, NoLink, RfPeak, AfPeak]`.

- type: Read only
- value: String
- example: `{"rx2":{"warnings":null}} -> {"rx2":{"warnings":["Aes256Error"]}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - options:
        - ▷ Aes256Error
        - ▷ LowSignal
        - ▷ NoLink
        - ▷ RfPeak
        - ▷ AfPeak
    - subscr: true

# "/rx2/name"

User-settable channel name for receiver channel 2.

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - length: 8
    - subscr: true

# "/rx2/mute"

Set audio output mute on/off of receiver channel 2.

- type: Read/Write
- value: boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx2/mates"

Information about active mates of the receiver channel 2 stored as array of string values.

- type: Read only
- value: String
- example: `{"rx2":{"mates":null}} -> {"rx2":{["mates":"mates/tx2"]}}`
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/rx2/gain"

Gain level for receiver channel 2. Valid values in range `[-3dB..+42dB]`, increment step = 3dB.

- type: Read/Write
- value: Number
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 42
    - min: -3
    - inc: 3
    - subscr: true

# "/rx2/frequency"

Carrier Frequency in kHz for receiver channel 2.

- type: Read/Write
- value: Number.
- example: `{"rx2":{"frequency":null/520000}}`
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: kHz
    - max: 1999000
    - min: 470200
    - inc: 25
    - subscr: true

# "/rx2/audio"

Information about active audio inputs/outputs of the receiver channel 2 stored as array of string values.

- type: Read only
- value: String
- example: `{"rx2":{"audio":null}} -> {"rx2":{"audio":["audio1/out2"]}}`
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/rx2/restore"

Command to restore device audio default settings of receiver channel 2. Valid options: "AUDIO_DEFAULTS" (restore audio defaults only).

- type: write-only
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ AUDIO_DEFAULTS = Restore audio defaults only

# "/rx3/identification/visual"

EW-DX EM 4 Dante exclusive! Command to start/stop receiver channel 3 identification visual (true: start; false: stop; null: read status).

- type: Read/Write
- value: Boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/presets/user/0"

EW-DX EM 4 Dante exclusive! set/get list of user frequencies `[kHz]` for receiver channel 3.

- type: Read/Write
- value: Number.
- example: `{"rx3":{"presets":{"user":{"0":[549200, 549800]}}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: true
    - count: -1
    - subscr: true

# "/rx3/presets/active"

EW-DX EM 4 Dante exclusive! set/get frequency from specific (presets) frequency list for receiver channel 3.

- type: Read/Write
- value: Number.
- example: select from factory list: index=0, channel=2: `{"rx3":{"presets":{"active":"factory:0:2"}}}` or select from user list: index=0, channel=1: `{"rx3":{"presets":{"active":"user:0:1"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/sync_settings/trim_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply trim setting on TX sync action for receiver channel 3.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/sync_settings/trim"

EW-DX EM 4 Dante exclusive! TX trim for receivder channel 3.

- type: Read/Write
- value: Number
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 6
    - min: -12
    - inc: 1
    - subscr: true

# "/rx3/sync_settings/name_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply name setting on TX sync action for receiver channel 3.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/sync_settings/mute_config_ts"

EW-DX EM 4 Dante exclusive! TS mute configuration (mute mode) for receiver channel 3.

- type: Read/Write
- value: string
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ af_mute = AF Mute
        - ▷ off = Disabled
        - ▷ push_to_talk = Push to talk
        - ▷ push_to_mute = Push to mute
    - subscr: true

# "/rx3/sync_settings/mute_config_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply mute configuration (mute mode) setting on TX sync action for receiver channel 3.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/sync_settings/mute_config"

EW-DX EM 4 Dante exclusive! TX mute configuration (mute mode) for receiver channel 3.

- type: Read/Write
- value: string
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off = Disabled
        - ▷ rf_mute = RF Mute
        - ▷ af_mute = AF Mute
    - subscr: true

# "/rx3/sync_settings/lowcut_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply low cut frequency setting on TX sync action for receiver channel 3.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/sync_settings/lowcut"

EW-DX EM 4 Dante exclusive! TX low cut frequency for receiver channel 3.

- type: Read/Write
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off
        - ▷ 30 Hz
        - ▷ 60 Hz
        - ▷ 80 Hz
        - ▷ 100 Hz
        - ▷ 120 Hz
    - subscr: true

# "/rx3/sync_settings/lock_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply Auto Lock setting on TX sync action for receiver channel 3.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/sync_settings/lock"

EW-DX EM 4 Dante exclusive! TX Autolock off/on for receiver channel 3.

- type: Read/Write
- value: Boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/sync_settings/led_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply LED setting on TX sync action for receiver channel 3.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/sync_settings/led"

EW-DX EM 4 Dante exclusive! TX LED off/on for receiver channel 3.

- type: Read/Write
- value: Boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/sync_settings/frequency_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply frequency setting on TX sync action for receiver channel 3.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/sync_settings/cable_emulation_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply cable emulation setting on TX sync action for receiver channel 3.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/sync_settings/cable_emulation"

EW-DX EM 4 Dante exclusive! TX cable emulation for receiver channel 3.

- type: Read/Write
- value: string
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off
        - ▷ type1
        - ▷ type2
        - ▷ type3
    - subscr: true

# "/rx3/warnings"

EW-DX EM 4 Dante exclusive! Active warnings on receiver channel 3 stored in an array of strings. Warnings: `[Aes256Error, LowSignal, NoLink, RfPeak, AfPeak]`.

- type: Read only
- value: String
- example: `{"rx3":{"warnings":null}} -> {"rx3":{"warnings":["Aes256Error"]}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - options:
        - ▷ Aes256Error
        - ▷ LowSignal
        - ▷ NoLink
        - ▷ RfPeak
        - ▷ AfPeak
    - subscr: true

# "/rx3/name"

EW-DX EM 4 Dante exclusive! User-settable channel name for receiver channel 3.

- type: Read/Write
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - length: 8
    - subscr: true

# "/rx3/mute"

EW-DX EM 4 Dante exclusive! Set audio output mute on/off of receiver channel 3.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx3/mates"

EW-DX EM 4 Dante exclusive! Information about active mates of the receiver channel 3 stored as array of string values.

- type: Read only
- value: String
- example: `{"rx3":{"mates":null}} -> {"rx3":{["mates":"mates/tx3"]}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/rx3/gain"

EW-DX EM 4 Dante exclusive! Gain level for receiver channel 3. Valid values in range `[-3dB..+42dB]`, increment step = 3dB.

- type: Read/Write
- value: Number
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 42
    - min: -3
    - inc: 3
    - subscr: true

# "/rx3/frequency"

EW-DX EM 4 Dante exclusive! Carrier Frequency in kHz for receiver channel 3.

- type: Read/Write
- value: Number.
- example: `{"rx3":{"frequency":null/520000}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: kHz
    - max: 1999000
    - min: 470200
    - inc: 25
    - subscr: true

# "/rx3/audio"

EW-DX EM 4 Dante exclusive! Information about active audio inputs/outputs of the receiver channel 3 stored as array of string values.

- type: Read only
- value: String
- example: `{"rx3":{"audio":null}} -> {"rx3":{"audio":["audio1/out3"]}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/rx3/restore"

EW-DX EM 4 Dante exclusive! Command to restore device audio default settings of receiver channel 3. Valid options: "AUDIO_DEFAULTS" (restore audio defaults only).

- type: write-only
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ AUDIO_DEFAULTS = Restore audio defaults only

# "/rx4/identification/visual"

EW-DX EM 4 Dante exclusive! Command to start/stop receiver channel 4 identification visual (true: start; false: stop; null: read status).

- type: Read/Write
- value: Boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/presets/user/0"

EW-DX EM 4 Dante exclusive! set/get list of user frequencies `[kHz]` for receiver channel 4.

- type: Read/Write
- value: Number.
- example: `{"rx4":{"presets":{"user":{"0":[548000, 548300, 548600]}}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: true
    - count: -1
    - subscr: true

# "/rx4/presets/active"

EW-DX EM 4 Dante exclusive! set/get frequency from specific (presets) frequency list for receiver channel 4.

- type: Read/Write
- value: Number.
- example: select from factory list: index=0, channel=2: `{"rx4":{"presets":{"active":"factory:0:2"}}}` or select from user list: index=0, channel=1: `{"rx4":{"presets":{"active":"user:0:1"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/sync_settings/trim_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply trim setting on TX sync action for receiver channel 4.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/sync_settings/trim"

EW-DX EM 4 Dante exclusive! TX trim for receivder channel 4.

- type: Read/Write
- value: Number
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 6
    - min: -12
    - inc: 1
    - subscr: true

# "/rx4/sync_settings/name_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply name setting on TX sync action for receiver channel 4.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/sync_settings/mute_config_ts"

TS mute configuration (mute mode) for receiver channel 4.

- type: Read/Write
- value: string
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ af_mute = AF Mute
        - ▷ off = Disabled
        - ▷ push_to_talk = Push to talk
        - ▷ push_to_mute = Push to mute
    - subscr: true

# "/rx4/sync_settings/mute_config_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply mute configuration (mute mode) setting on TX sync action for receiver channel 4.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/sync_settings/mute_config"

EW-DX EM 4 Dante exclusive! TX mute configuration (mute mode) for receiver channel 4.

- type: Read/Write
- value: string
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off = Disabled
        - ▷ rf_mute = RF Mute
        - ▷ af_mute = AF Mute
    - subscr: true

# "/rx4/sync_settings/lowcut_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply low cut frequency setting on TX sync action for receiver channel 4.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/sync_settings/lowcut"

EW-DX EM 4 Dante exclusive! TX low cut frequency for receiver channel 4.

- type: Read/Write
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off
        - ▷ 30 Hz
        - ▷ 60 Hz
        - ▷ 80 Hz
        - ▷ 100 Hz
        - ▷ 120 Hz
    - subscr: true

# "/rx4/sync_settings/lock_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply Auto Lock setting on TX sync action for receiver channel 4.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/sync_settings/lock"

EW-DX EM 4 Dante exclusive! TX Autolock off/on for receiver channel 4.

- type: Read/Write
- value: Boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/sync_settings/led_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply LED setting on TX sync action for receiver channel 4.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/sync_settings/led"

EW-DX EM 4 Dante exclusive! TX LED off/on for receiver channel 4.

- type: Read/Write
- value: Boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/sync_settings/frequency_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply frequency setting on TX sync action for receiver channel 4.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/sync_settings/cable_emulation_ignore"

EW-DX EM 4 Dante exclusive! Skip/apply cable emulation setting on TX sync action for receiver channel 4

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/sync_settings/cable_emulation"

EW-DX EM 4 Dante exclusive! TX cable emulation for receiver channel 4.

- type: Read/Write
- value: string
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ off
        - ▷ type1
        - ▷ type2
        - ▷ type3
    - subscr: true

# "/rx4/warnings"

EW-DX EM 4 Dante exclusive! Active warnings on receiver channel 4 stored in an array of strings. Warnings: `[Aes256Error, LowSignal, NoLink, RfPeak, AfPeak]`.

- type: Read only
- value: String
- example: `{"rx4":{"warnings":null}} -> {"rx4":{"warnings":["Aes256Error"]}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - options:
        - ▷ Aes256Error
        - ▷ LowSignal
        - ▷ NoLink
        - ▷ RfPeak
        - ▷ AfPeak
    - subscr: true

# "/rx4/name"

EW-DX EM 4 Dante exclusive! User-settable channel name for receiver channel 4.

- type: Read/Write
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - length: 8
    - subscr: true

# "/rx4/mute"

EW-DX EM 4 Dante exclusive! Set audio output mute on/off of receiver channel 4.

- type: Read/Write
- value: boolean
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/rx4/mates"

EW-DX EM 4 Dante exclusive! Information about active mates of the receiver channel 4 stored as array of string values.

- type: Read only
- value: String
- example: `{"rx4":{"mates":null}} -> {"rx4":{"mates":["mates/tx4"]}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/rx4/gain"

EW-DX EM 4 Dante exclusive! Gain level for receiver channel 4. Valid values in range `[-3dB..+42dB]`, increment step = 3dB.

- type: Read/Write
- value: Number
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 42
    - min: -3
    - inc: 3
    - subscr: true

# "/rx4/frequency"

EW-DX EM 4 Dante exclusive! Carrier Frequency in kHz for receiver channel 4.

- type: Read/Write
- value: Number.
- example: `{"rx4":{"frequency":null/520000}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: kHz
    - max: 1999000
    - min: 470200
    - inc: 25
    - subscr: true

# "/rx4/audio"

EW-DX EM 4 Dante exclusive! Information about active audio inputs/outputs of the receiver channel 4 stored as array of string values.

- type: Read only
- value: String
- example: `{"rx4":{"audio":null}} -> {"rx4":{"audio":["audio1/out4"]}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/rx4/restore"

EW-DX EM 4 Dante exclusive! Command to restore device audio default settings of receiver channel 4. Valid options: "AUDIO_DEFAULTS" (restore audio defaults only).

- type: write-only
- value: String
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ AUDIO_DEFAULTS = Restore audio defaults only

# "/audio1/out1/level"

Af output level for receiver channel 1. Valid values in range `[-24dB..+18dB]`, increment step = 6dB.

- type: Read/Write
- value: Number
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 18
    - min: -24
    - inc: 6
    - subscr: true

# "/audio1/out2/level"

Af output level for receiver channel 2. Valid values in range `[-24dB..+18dB]`, increment step = 6dB.

- type: Read/Write
- value: Number
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 18
    - min: -24
    - inc: 6
    - subscr: true

# "/audio1/out3/level"

EW-DX EM 4 Dante exclusive! Af output level for receiver channel 3. Valid values in range `[-24dB..+18dB]`, increment step = 6dB.

- type: Read/Write
- value: Number
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 18
    - min: -24
    - inc: 6
    - subscr: true

# "/audio1/out4/level"

EW-DX EM 4 Dante exclusive! Af output level for receiver channel 4. Valid values in range -24dB..+18dB], increment step = 6dB.

- type: Read/Write
- value: Number
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: dB
    - max: 18
    - min: -24
    - inc: 6
    - subscr: true

# "/m/rx1/rssi"

Type: double (ro). Value range: -107.0..0.0 `[dBm]`. Returns RF radio signal strength indicator (RSSI) for receiver channel 1.

- example: `{"m":{"rx2":{"rssi":null}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dBm
    - max: 0
    - min: -107
    - subscr: true

# "/m/rx1/rsqi"

Type: number (ro). Value range: 0..100 `[%]`, Returns RF signal quality indicator for receiver channel 1.

- example: `{"m":{"rx1":{"rsqi":null}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: %
    - max: 100
    - min: 0
    - inc: 1
    - subscr: true

# "/m/rx1/divi"

Type: number (ro). Returns RF diversity indicator for receiver channel 1. 0 is for None; 1 is for antenna input A; 2 is for antenna input B.

- example: `{"m":{"rx1":{"divi":null}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - max: 2
    - min: 0
    - inc: 1
    - options:
        - ▷ 0 = None
        - ▷ 1 = Antenna input A
        - ▷ 2 = Antenna input B
    - subscr: true

# "/m/rx1/af"

Type: double (ro). Value range : -138.5..0.0 `[dBfs]`, Returns audio level for receiver channel 1.

- example: `{"m":{"rx1":{"af":null}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dBfs
    - max: 0
    - min: -138.5
    - inc: 0.1
    - subscr: true

# "/m/rx2/rssi"

Type: double (ro). Value range: -107.0..0.0 `[dBm]`. Returns RF radio signal strength indicator (RSSI) for receiver channel 2.

- example: `{"m":{"rx2":{"rssi":null}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dBm
    - max: 0
    - min: -107
    - subscr: true

# "/m/rx2/rsqi"

Type: number (ro). Value range: 0..100 `[%]`, Returns RF signal quality indicator for receiver channel 2.

- example: `{"m":{"rx2":{"rsqi":null}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: %
    - max: 100
    - min: 0
    - inc: 1
    - subscr: true

# "/m/rx2/divi"

Type: number (ro). Returns RF diversity indicator for receiver channel 2. 0 is for None; 1 is for antenna input A; 2 is for antenna input B.

- example: `{"m":{"rx2":{"divi":null}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - max: 2
    - min: 0
    - inc: 1
    - options:
        - ▷ 0 = None
        - ▷ 1 = Antenna input A
        - ▷ 2 = Antenna input B
    - subscr: true

# "/m/rx2/af"

Type: double (ro). Value range : -138.5..0.0 `[dBfs]`, Returns audio level for receiver channel 2.

- example: `{"m":{"rx2":{"af":null}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dBfs
    - max: 0
    - min: -138.5
    - inc: 0.1
    - subscr: true

# "/m/rx3/rssi"

EW-DX EM 4 Dante exclusive! Type: double (ro). Value range: -107.0..0.0 `[dBm]`. Returns RF radio signal strength indicator (RSSI) for receiver channel 3.

- example: `{"m":{"rx3":{"rssi":null}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dBm
    - max: 0
    - min: -107
    - subscr: true

# "/m/rx3/rsqi"

EW-DX EM 4 Dante exclusive! Type: number (ro). Value range: 0..100 `[%]`, Returns RF signal quality indicator for receiver channel 3.

- example: `{"m":{"rx3":{"rsqi":null}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: %
    - max: 100
    - min: 0
    - inc: 1
    - subscr: true

# "/m/rx3/divi"

EW-DX EM 4 Dante exclusive! Type: number (ro). Returns RF diversity indicator for receiver channel 3. 0 is for None; 1 is for antenna input A; 2 is for antenna input B.

- example: `{"m":{"rx3":{"divi":null}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - max: 2
    - min: 0
    - inc: 1
    - options:
        - ▷ 0 = None
        - ▷ 1 = Antenna input A
        - ▷ 2 = Antenna input B
    - subscr: true

# "/m/rx3/af"

EW-DX EM 4 Dante exclusive! Type: double (ro). Value range : -138.5..0.0 `[dBfs]`, Returns audio level for receiver channel 3.

- example: `{"m":{"rx3":{"af":null}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dBfs
    - max: 0
    - min: -138.5
    - inc: 0.1
    - subscr: true

# "/m/rx4/rssi"

EW-DX EM 4 Dante exclusive! Type: double (ro). Value range: -107.0..0.0 `[dBm]`. Returns RF radio signal strength indicator (RSSI) for receiver channel 4.

- example: `{"m":{"rx4":{"rssi":null}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dBm
    - max: 0
    - min: -107
    - subscr: true

# "/m/rx4/rsqi"

EW-DX EM 4 Dante exclusive! Type: number (ro). Value range: 0..100 `[%]`, Returns RF signal quality indicator for receiver channel 4.

- example: `{"m":{"rx4":{"rsqi":null}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: %
    - max: 100
    - min: 0
    - inc: 1
    - subscr: true

# "/m/rx4/divi"

EW-DX EM 4 Dante exclusive! Type: number (ro). Returns RF diversity indicator for receiver channel 4. 0 is for None; 1 is for antenna input A; 2 is for antenna input B.

- example: `{"m":{"rx4":{"divi":null}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - max: 2
    - min: 0
    - inc: 1
    - options:
        - ▷ 0 = None
        - ▷ 1 = Antenna input A
        - ▷ 2 = Antenna input B
    - subscr: true

# "/m/rx4/af"

EW-DX EM 4 Dante exclusive! Type: double (ro). Value range : -138.5..0.0 `[dBfs]`, Returns audio level for receiver channel 4.

- example: `{"m":{"rx4":{"af":null}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dBfs
    - max: 0
    - min: -138.5
    - inc: 0.1
    - subscr: true

# "/mates/tx1/battery/type"

TX battery type info got from receiver channel 1 stored as string. Possible values: "Battery", "Primary Cell". Replied error code 424 indicates that TX is not present or doesn't send valid battery information.

- type: Read only
- value: string
- example: `{"mates":{"tx1":{"battery":{"type":null}}}} -> {"mates":{"tx1":{"battery":{"type":"Battery"}}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx1/battery/lifetime"

TX battery lifetime in minutes got from receiver channel 1 stored as number. Replied error code 424 indicates that TX is not present or doesn't send valid battery lifetime.

- type: Read only
- value: int
- example: `{"mates":{"tx1":{"battery":{"lifetime":null}}}} -> {"mates":{"tx1":{"battery":{"lifetime":312}}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx1/battery/gauge"

TX battery gauge (charge in percent) got from receiver channel 1 stored as number. Replied error code 424 indicates that TX is not present or doesn't send valid battery gauge.

- type: Read only
- value: int
- example: `{"mates":{"tx1":{"battery":{"gauge":null}}}} -> {"mates":{"tx1":{"battery":{"gauge":65}}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: %
    - max: 100
    - min: 0
    - inc: 1
    - subscr: true

# "/mates/tx1/warnings"

TX warnings occuring on receiver active link channel 1 stored in an array of strings. Warnings: `[AfPeak, LowBattery]`.

- type: Read only
- value: String
- example: `{"mates":{"tx1":{"warnings":null}}} -> {"mates":{"tx1":{"warnings":["LowBattery","AfPeak"]}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - options:
        - ▷ AfPeak
        - ▷ LowBattery
    - subscr: true

# "/mates/tx1/version"

TX SW version information got from receiver channel 1 stored as a string . Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"mates":{"tx1":{"version":null}}} -> {"mates":{"tx1":{"version":"1.2.3"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx1/type"

TX type information got from receiver channel 1 stored as a string . Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"mates":{"tx1":{"type":null}}} -> {"mates":{"tx1":{"type":"SKMPLUS"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx1/trim"

TX trim information got from receiver channel 1 stored as a number . Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: number
- example: `{"mates":{"tx1":{"trim":null}}} -> {"mates":{"tx1":{"trim":-12}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dB
    - max: 6
    - min: -12
    - inc: 1
    - subscr: true

# "/mates/tx1/name"

TX name information got from receiver channel 1 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx1":{"name":null}}}]}}} -> {"mates":{"tx1": {"name":"EWDX-CH1"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx1/mute_config_ts"

TS mute configuration (mute mode) information got from receiver channel 1 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx1":{"mute_config_ts":null}}}]}}} -> {"mates": {"tx1":{"mute_config_ts":"push_to_talk"}}}`
- limits
  - type: String
  - const: false
  - writeable: false
  - options:
    - ▷ af_mute = AF Mute
    - ▷ off = Disabled
    - ▷ push_to_talk = Push to talk
    - ▷ push_to_mute = Push to mute
  - subscr: true

# "/mates/tx1/mute_config"

TX mute configuration (mute mode) information got from receiver channel 1 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx1":{"mute_config":null}}}]}}} -> {"mates":{"tx1":{"mute_config":"af_mute"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - options:
        - ▷ off = Disabled
        - ▷ rf_mute = RF Mute
        - ▷ af_mute = AF Mute
    - subscr: true

# "/mates/tx1/mute"

TX Mute Switch On/Off information got from receiver channel 1 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx1":{"mute":null}}}]}}} -> {"mates":{"tx1":{"mute":true}}}`
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx1/lowcut"

TX lowcut information got from receiver channel 1 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information. Valid strings: "off", "30 Hz", "60 Hz", ...

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx1":{"lowcut":null}}}]}}} -> {"mates":{"tx1": {"lowcut":"60 Hz"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx1/lock"

TX auto lock information got from receiver channel 1 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx1":{"lock":null}}}]}}} -> {"mates":{"tx1": {"lock":true}}}`
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx1/led"

TX LED information got from receiver channel 1 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx1":{"led":null}}}]}}} -> {"mates":{"tx1": {"led":true}}}`
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx1/identification"

TX identification information got from receiver channel 1 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx1":{"identification":null}}}]}}} -> {"mates": {"tx1":{"identification":false}}}`
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx1/capsule"

TX capsule information got from receiver channel 1 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information. Valid strings: "unknown", "KK 205", "KK 204", "ME 9002", "ME 9004", "ME 9005", "MME 865", "MD 9235", "MMD 945", "MMD 935", "MMD 845", "MMD 835", "MMD 815-1", "MMK 965", "MMD 42-I"

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx1":{"capsule":null}}}]}}} -> {"mates":{"tx1":{"capsule":"MMD 935"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx1/cable_emulation"

TX cable emulation information got from receiver channel 1 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx1":{"cable_emulation":null}}}]}}} -> {"mates": {"tx1":{"cable_emulation":"type2"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - options:
        - ▷ off
        - ▷ type1
        - ▷ type2
        - ▷ type3
    - subscr: true

# "/mates/tx2/battery/type"

TX battery type info got from receiver channel 2 stored as string. Possible values: "Battery", "Primary Cell". Replied error code 424 indicates that TX is not present or doesn't send valid battery information.

- type: Read only
- value: string
- example: `{"mates":{"tx2":{"battery":{"type":null}}}} -> {"mates":{"tx2":{"battery":{"type":"Battery"}}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx2/battery/lifetime"

TX battery lifetime in minutes got from receiver channel 2 stored as number. Replied error code 424 indicates that TX is not present or doesn't send valid battery lifetime.

- type: Read only
- value: int
- example: `{"mates":{"tx2":{"battery":{"lifetime":null}}}} -> {"mates":{"tx2":{"battery":{"lifetime":312}}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx2/battery/gauge"

TX battery gauge (charge in percent) got from receiver channel 2 stored as number. Replied error code 424 indicates that TX is not present or doesn't send valid battery gauge.

- type: Read only
- value: int
- example: `{"mates":{"tx2":{"battery":{"gauge":null}}}} -> {"mates":{"tx2":{"battery":{"gauge":80}}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: %
    - max: 100
    - min: 0
    - inc: 1
    - subscr: true

# "/mates/tx2/warnings"

TX occuring on receiver active link channel 2 stored in an array of strings. Warnings: `[AfPeak, LowBattery]`.

- type: Read only
- value: String
- example: `{"mates":{"tx2":{"warnings":null}}} -> {"mates":{"tx2":{"warnings":}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - options:
        - ▷ AfPeak
        - ▷ LowBattery
    - subscr: true

# "/mates/tx2/version"

TX SW version information got from receiver channel 2 stored as a string . Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"mates":{"tx2":{"version":null}}} -> {"mates":{"tx2":{"version":"1.2.3"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx2/type"

TX type information got from receiver channel 2 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"mates":{"tx2":{"type":null}}} -> {"mates":{"tx2":{"type":"SKM"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx2/trim"

TX trim information got from receiver channel 2 stored as a number . Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: number
- example: `{"mates":{"tx2":{"trim":null}}} -> {"mates":{"tx2":{"trim":-12}}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dB
    - max: 6
    - min: -12
    - inc: 1
    - subscr: true

# "/mates/tx2/name"

TX name information got from receiver channel 2 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx2":{"name":null}}}]}}} -> {"mates":{"tx2":{"name":"EWDX-CH2"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx2/mute_config_ts"

TS mute configuration (mute mode) information got from receiver channel 2 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx2":{"mute_config_ts":null}}}]}}} -> {"mates": {"tx2":{"mute_config_ts":"push_to_mute"}}}`
- limits
  - type: String
  - const: false
  - writeable: false
  - options:
    - ▷ af_mute = AF Mute
    - ▷ off = Disabled
    - ▷ push_to_talk = Push to talk
    - ▷ push_to_mute = Push to mute
  - subscr: true

# "/mates/tx2/mute_config"

TX mute configuration (mute mode) information got from receiver channel 2 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx2":{"mute_config":null}}}]}}} -> {"mates":{"tx2":{"mute_config":"af_mute"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - options:
        - ▷ off = Disabled
        - ▷ rf_mute = RF Mute
        - ▷ af_mute = AF Mute
    - subscr: true

# "/mates/tx2/mute"

TX Mute Switch On/Off information got from receiver channel 2 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx2":{"mute":null}}}]}}} -> {"mates":{"tx2":{"mute":true}}}`
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx2/lowcut"

TX lowcut information got from receiver channel 2 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information. Valid strings: "off", "30 Hz", "60 Hz", ...

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx2":{"lowcut":null}}}]}}} -> {"mates":{"tx2":{"lowcut":"60 Hz"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx2/lock"

TX auto lock information got from receiver channel 2 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx2":{"lock":null}}}]}}} -> {"mates":{"tx2":{"lock":true}}}`
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx2/led"

TX LED information got from receiver channel 2 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx2":{"led":null}}}]}}} -> {"mates":{"tx2":{"led":true}}}`
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx2/identification"

TX identification information got from receiver channel 2 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx2":{"identification":null}}}]}}} -> {"mates": {"tx2":{"identification":true}}}`
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx2/capsule"

TX capsule information got from receiver channel 2 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information. Valid strings: "unknown", "KK 205", "KK 204", "ME 9002", "ME 9004", "ME 9005", "MME 865", "MD 9235", "MMD 945", "MMD 935", "MMD 845", "MMD 835", "MMD 815-1", "MMK 965", "MMD 42-I"

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx2":{"capsule":null}}}]}}} -> {"mates":{"tx2":{"capsule":"MMD 935"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx2/cable_emulation"

TX cable emulation information got from receiver channel 2 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx2":{"cable_emulation":null}}}]}}} -> {"mates": {"tx2":{"cable_emulation":"type2"}}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - options:
        - ▷ off
        - ▷ type1
        - ▷ type2
        - ▷ type3
    - subscr: true

# "/mates/tx3/battery/type"

EW-DX EM 4 Dante exclusive! TX battery type info got from receiver channel 3 stored as string. Possible values: "Battery", "Primary Cell". Replied error code 424 indicates that TX is not present or doesn't send valid battery information.

- type: Read only
- value: string
- example: `{"mates":{"tx3":{"battery":{"type":null}}}} -> {"mates":{"tx3":{"battery":{"type":"Battery"}}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx3/battery/lifetime"

EW-DX EM 4 Dante exclusive! TX battery lifetime in minutes got from receiver channel 3 stored as number. Replied error code 424 indicates that TX is not present or doesn't send valid battery lifetime.

- type: Read only
- value: int
- example: `{"mates":{"tx3":{"battery":{"lifetime":null}}}} -> {"mates":{"tx3":{"battery":{"lifetime":312}}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx3/battery/gauge"

EW-DX EM 4 Dante exclusive! TX battery gauge (charge in percent) got from receiver channel 3 stored as number. Replied error code 424 indicates that TX is not present or doesn't send valid battery gauge.

- type: Read only
- value: int
- example: `{"mates":{"tx3":{"battery":{"gauge":null}}}} -> {"mates":{"tx3":{"battery":{"gauge":80}}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: %
    - max: 100
    - min: 0
    - inc: 1
    - subscr: true

# "/mates/tx3/warnings"

EW-DX EM 4 Dante exclusive! TX occuring on receiver active link channel 3 stored in an array of strings. Warnings: .

- type: Read only
- value: String
- example: `{"mates":{"tx3":{"warnings":null}}} -> {"mates":{"tx3":{"warnings":}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - options:
        - ▷ AfPeak
        - ▷ LowBattery
    - subscr: true

# "/mates/tx3/version"

EW-DX EM 4 Dante exclusive! TX SW version information got from receiver channel 3 stored as a string . Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"mates":{"tx3":{"version":null}}} -> {"mates":{"tx3":{"version":"1.2.3"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx3/type"

EW-DX EM 4 Dante exclusive! TX type information got from receiver channel 3 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"mates":{"tx3":{"type":null}}} -> {"mates":{"tx3":{"type":"SKM"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx3/trim"

EW-DX EM 4 Dante exclusive! TX trim information got from receiver channel 3 stored as a number . Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: number
- example: `{"mates":{"tx3":{"trim":null}}} -> {"mates":{"tx3":{"trim":-12}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dB
    - max: 6
    - min: -12
    - inc: 1
    - subscr: true

# "/mates/tx3/name"

EW-DX EM 4 Dante exclusive! TX name information got from receiver channel 3 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx3":{"name":null}}}]}}} -> {"mates":{"tx3": {"name":"EWDX-CH3"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx3/mute_config_ts"

EW-DX EM 4 Dante exclusive! TS mute configuration (mute mode) information got from receiver channel 3 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx3":{"mute_config_ts":null}}}]}}} -> {"mates": {"tx3":{"mute_config_ts":"push_to_mute"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - options:
        - ▷ af_mute = AF Mute
        - ▷ off = Disabled
        - ▷ push_to_talk = Push to talk
        - ▷ push_to_mute = Push to mute
    - subscr: true

# "/mates/tx3/mute_config"

EW-DX EM 4 Dante exclusive! TX mute configuration (mute mode) information got from receiver channel 3 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx3":{"mute_config":null}}}]}}} -> {"mates":{"tx3":{"mute_config":"af_mute"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - options:
        - ▷ off = Disabled
        - ▷ rf_mute = RF Mute
        - ▷ af_mute = AF Mute
    - subscr: true

# "/mates/tx3/mute"

EW-DX EM 4 Dante exclusive! TX Mute Switch On/Off information got from receiver channel 3 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx3":{"mute":null}}}]}}} -> {"mates":{"tx3":{"mute":true}}}`
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx3/lowcut"

EW-DX EM 4 Dante exclusive! TX lowcut information got from receiver channel 3 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information. Valid strings: "off", "30 Hz", "60 Hz", ...

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx3":{"lowcut":null}}}]}}} -> {"mates":{"tx3":{"lowcut":"60 Hz"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx3/lock"

EW-DX EM 4 Dante exclusive! TX auto lock information got from receiver channel 3 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx3":{"lock":null}}}]}}} -> {"mates":{"tx3":{"lock":true}}}`
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx3/led"

EW-DX EM 4 Dante exclusive! TX LED information got from receiver channel 3 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx3":{"led":null}}}]}}} -> {"mates":{"tx3":{"led":true}}}`
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx3/identification"

EW-DX EM 4 Dante exclusive! TX identification information got from receiver channel 3 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx3":{"identification":null}}}]}}} -> {"mates":{"tx3":{"identification":true}}}`
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx3/capsule"

EW-DX EM 4 Dante exclusive! TX capsule information got from receiver channel 3 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information. Valid strings: "unknown", "KK 205", "KK 204", "ME 9002", "ME 9004", "ME 9005", "MME 865", "MD 9235", "MMD 945", "MMD 935", "MMD 845", "MMD 835", "MMD 815-1", "MMK 965", "MMD 42-I"

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx3":{"capsule":null}}}]}}} -> {"mates":{"tx3":{"capsule":"MMD 935"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx3/cable_emulation"

EW-DX EM 4 Dante exclusive! TX cable emulation information got from receiver channel 3 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx3":{"cable_emulation":null}}}]}}} -> {"mates":{"tx3":{"cable_emulation":"type2"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - options:
        - ▷ off
        - ▷ type1
        - ▷ type2
        - ▷ type3
    - subscr: true

# "/mates/tx4/battery/type"

EW-DX EM 4 Dante exclusive! TX battery type info got from receiver channel 4 stored as string. Possible values: "Battery", "Primary Cell". Replied error code 424 indicates that TX is not present or doesn't send valid battery information.

- type: Read only
- value: string
- example: `{"mates":{"tx4":{"battery":{"type":null}}}} -> {"mates":{"tx4":{"battery":{"type":"Battery"}}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx4/battery/lifetime"

EW-DX EM 4 Dante exclusive! TX battery lifetime in minutes got from receiver channel 4 stored as number. Replied error code 424 indicates that TX is not present or doesn't send valid battery lifetime.

- type: Read only
- value: int
- example: `{"mates":{"tx4":{"battery":{"lifetime":null}}}} -> {"mates":{"tx4":{"battery":{"lifetime":312}}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx4/battery/gauge"

EW-DX EM 4 Dante exclusive! TX battery gauge (charge in percent) got from receiver channel 4 stored as number. Replied error code 424 indicates that TX is not present or doesn't send valid battery gauge.

- type: Read only
- value: int
- example: `{"mates":{"tx4":{"battery":{"gauge":null}}}} -> {"mates":{"tx4":{"battery":{"gauge":80}}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: %
    - max: 100
    - min: 0
    - inc: 1
    - subscr: true

# "/mates/tx4/warnings"

EW-DX EM 4 Dante exclusive! TX occuring on receiver active link channel 4 stored in an array of strings. Warnings: .

- type: Read only
- value: String
- example: `{"mates":{"tx4":{"warnings":null}}} -> {"mates":{"tx4":{"warnings":}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - options:
        - ▷ AfPeak
        - ▷ LowBattery
    - subscr: true

# "/mates/tx4/version"

EW-DX EM 4 Dante exclusive! TX SW version information got from receiver channel 4 stored as a string . Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"mates":{"tx4":{"version":null}}} -> {"mates":{"tx4":{"version":"1.2.3"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx4/type"

EW-DX EM 4 Dante exclusive! TX type information got from receiver channel 4 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"mates":{"tx4":{"type":null}}} -> {"mates":{"tx4":{"type":"SKM"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx4/trim"

EW-DX EM 4 Dante exclusive! TX trim information got from receiver channel 4 stored as a number . Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: number
- example: `{"mates":{"tx4":{"trim":null}}} -> {"mates":{"tx4":{"trim":-12}}}`
- hidden
- visible under conditions
- limits
    - type: Number
    - const: false
    - writeable: false
    - units: dB
    - max: 6
    - min: -12
    - inc: 1
    - subscr: true

# "/mates/tx4/name"

EW-DX EM 4 Dante exclusive! TX name information got from receiver channel 4 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx4":{"name":null}}}]}}} -> {"mates":{"tx4": {"name":"EWDX-CH4"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx4/mute_config_ts"

EW-DX EM 4 Dante exclusive! TS mute configuration (mute mode) information got from receiver channel 4 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx4":{"mute_config_ts":null}}}]}}} -> {"mates": {"tx4":{"mute_config_ts":"push_to_mute"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - options:
        - ▷ af_mute = AF Mute
        - ▷ off = Disabled
        - ▷ push_to_talk = Push to talk
        - ▷ push_to_mute = Push to mute
    - subscr: true

# "/mates/tx4/mute_config"

EW-DX EM 4 Dante exclusive! TX mute configuration (mute mode) information got from receiver channel 4 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx4":{"mute_config":null}}}]}}} -> {"mates":{"tx4":{"mute_config":"af_mute"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - options:
        - ▷ off = Disabled
        - ▷ rf_mute = RF Mute
        - ▷ af_mute = AF Mute
    - subscr: true

# "/mates/tx4/mute"

EW-DX EM 4 Dante exclusive! TX Mute Switch On/Off information got from receiver channel 4 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx4":{"mute":null}}}]}}} -> {"mates":{"tx4":{"mute":true}}}`
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx4/lowcut"

EW-DX EM 4 Dante exclusive! TX lowcut information got from receiver channel 4 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information. Valid strings: "off", "30 Hz", "60 Hz", ...

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx4":{"lowcut":null}}}]}}} -> {"mates":{"tx4":{"lowcut":"60 Hz"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx4/lock"

EW-DX EM 4 Dante exclusive! TX auto lock information got from receiver channel 4 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx4":{"lock":null}}}]}}} -> {"mates":{"tx4":{"lock":true}}}`
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx4/led"

EW-DX EM 4 Dante exclusive! TX LED information got from receiver channel 4 stored as a bool. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx4":{"led":null}}}]}}} -> {"mates":{"tx4":{"led":true}}}`
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx4/identification"

EW-DX EM 4 Dante exclusive! TX identification information got from receiver channel 4 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: bool
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx4":{"identification":null}}}]}}} -> {"mates":{"tx4":{"identification":true}}}`
- hidden
- visible under conditions
- limits
    - type: Boolean
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx4/capsule"

EW-DX EM 4 Dante exclusive! TX capsule information got from receiver channel 4 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information. Valid strings: "unknown", "KK 205", "KK 204", "ME 9002", "ME 9004", "ME 9005", "MME 865", "MD 9235", "MMD 945", "MMD 935", "MMD 845", "MMD 835", "MMD 815-1", "MMK 965", "MMD 42-I"

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx4":{"capsule":null}}}]}}} -> {"mates":{"tx4":{"capsule":"MMD 935"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - subscr: true

# "/mates/tx4/cable_emulation"

EW-DX EM 4 Dante exclusive! TX cable emulation information got from receiver channel 4 stored as a string. Replied error code 424 indicates that TX is not present or doesn't send valid information.

- type: Read only
- value: string
- example: `{"osc":{"state":{"subscribe":{["#":{"min":0, "max":0,"count":1000, "lifetime":60},"mates":{"tx4":{"cable_emulation":null}}}]}}} -> {"mates":{"tx4":{"cable_emulation":"type2"}}}`
- hidden
- visible under conditions
- limits
    - type: String
    - const: false
    - writeable: false
    - options:
        - ▷ off
        - ▷ type1
        - ▷ type2
        - ▷ type3
    - subscr: true

# 9. SSC String characters

- ASCII 32...126 including escaped characters ASCII 34('"'), ASCII 47('/'), ASCII 92('\')
- Escaped primitives '\b', '\f', '\r' ,'\n', '\t'
- No unicode patterns '\uxxxx'

# 10. SSC Error List (EW-DX EM)

- 100 : continue
- 102 : processing
- 200 : OK
- 201 : created
- 202 : adapted
- 210 : partial success
- 310 : subscription terminates
- 400 : message not understood
- 401 : authorisation needed
- 403 : forbidden
- 404 : address not found
- 406 : not acceptable
- 408 : request time out
- 409 : conflict
- 410 : gone
- 413 : request too long
- 414 : request too complex
- 416 : requested range not satisfiable
- 422 : unprocessable entity
- 423 : locked
- 424 : failed dependency
- 450 : answer too long
- 454 : parameter address not found
- 500 : internal server error
- 501 : not implemented
- 503 : service unavailable

# 11. Developer’s Guide for CHG 70N-C

This chapter describes in detail how a developer should use the SSC interface as implemented for the CHG 70N-C or CHG 70N.

## Limitations

### SSC Transport Layer

The SSC Server implemented for EW-DX devices supports only UDP/IP as transport protocol. All the devices support IPv4.

### Subscriptions

The CHG 70N-C charger supports SSC Method subscriptions from up to eight different SSC clients simultaneously.

# 12. SSC Method List (CHG 70N-C)

## "/device/identification/visual"

Command to start/stop device identification visual (true: start; false: stop; null: read status).

- type: Read/Write
- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/device/identity/version"

Product version.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false

# "/device/identity/vendor"

Command replies "Sennheiser electronic SE & CO KG".

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false

# "/device/identity/serial"

Not yet implemented. Serial no from production.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false

# "/device/identity/product"

Product label

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false

# "/device/network/ether/macs"

List of MAC addresses.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/device/network/ether/interfaces"

List of all ethernet interfaces.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/device/network/ipv4/manual_netmask"

Static (Manual) Netmask like "255.255.255.0".

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - count: 1
    - subscr: true

# "/device/network/ipv4/manual_ipaddr"

Static (Manual) IPv4 addr like "192.168.1.203".

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - count: 1
    - subscr: true

# "/device/network/ipv4/manual_gateway"

Static (Manual) Gateway addr like "192.168.1.1".

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - count: 1
    - subscr: true

# "/device/network/ipv4/netmask"

Online Netmask like "255.255.255.0".

- type: Read-only
- value: String
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 1
    - subscr: true

# "/device/network/ipv4/ipaddr"

Online IPv4 addr like "192.168.1.203".

- type: Read-only
- value: String
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 1
    - subscr: true

# "/device/network/ipv4/interfaces"

List of ipv4 interface indices.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/device/network/ipv4/gateway"

Online Gateway addr like "192.168.1.1".

- type: Read-only
- value: String
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 1
    - subscr: true

# "/device/network/ipv4/auto"

Command to configure ip settings automatically (true: use DHCP and ZeroConf (Auto-IP); false: set ip address/netmask/gateway manually).

- type: Read/Write
- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/device/network/mdns"

Command to enable, disable or read mdns state.

- value: Boolean
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - count: 1
    - subscr: true

# "/device/warnings"

Command to read device warnings.

- type: Read
- value: String
- limits
    - type: String
    - const: false
    - writeable: false
    - length: 40
    - options: CascadeComError
    - subscr: true

# "/device/time"

Current absolute clock value (in seconds) of the device since 2000-01-01T00:00:00+0000 UTC.

- type: Read
- value: Integer
- limits
    - type: Number
    - const: false
    - writeable: true
    - units: s
    - max: -1
    - min: 0
    - inc: 1

# "/device/storage_mode"

Starts storage mode.

- type: Read/Write
- value: bool example: `{"device":{"storage_mode":true}}`
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - subscr: true

# "/device/restore"

Command to restore device default settings. Valid options: "FACTORY_DEFAULTS" (restore all factory defaults), "AUDIO_DEFAULTS" (restore audio defaults only).

- type: write-only
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - options:
        - ▷ "FACTORY_DEFAULTS" = restores all factory defaults
        - ▷ "AUDIO_DEFAULTS" = restore audio defaults only

# "/device/restart"

Restarts the device

- limits
    - type: Boolean
    - const: false
    - writeable: true

# "/device/name"

Command to set or read device name.

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - length: 18
    - subscr: true

# "/device/location"

Command to set or read device location.

- type: Read/Write
- value: String
- limits
    - type: String
    - const: false
    - writeable: true
    - length: 400
    - subscr: true

# "/device/language"

List of supported languages. English only = `["en_GB"]`.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false
    - count: 1

# "/device/cascade"

MAC addresses of cascade participants as array of strings.

- type: Read-only
- value: String
- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - subscr: true

# "/interface/version"

Command to read Win-Plus SSC interface version.

- type: Read-only
- value: String
- limits
    - type: String
    - const: true
    - writeable: false

# "/osc/state/auth/access"

Read authentication access of current SSC client. Response value is an array of strings, comma separated

- limits
    - type: String
    - const: false
    - writeable: false
    - count: -1
    - subscr: true

## "/osc/state/prettyprint"

SSC reply output style is not supported. Returns false.

- limits (hidden)

# "/osc/state/close"

SSC connection close.

- limits (hidden)

# "/osc/state/subscribe"

SSC subscriptions.

- limits (hidden)

# "/osc/feature/timetag"

SSC timed method execution is not supported. Returns false.

- limits (hidden)

# "/osc/feature/baseaddr"

SSC interactive method address base is not supported. Returns false.

- limits (hidden)

# "/osc/feature/subscription"

SSC subscriptions are supported. Returns true.

- limits (hidden)

# "/osc/feature/pattern"

SSC message dispatching and pattern matching are supported. Returns "*?".

- limits (hidden)

# "/osc/limits"

SSC method parameter range reflection.

- limits (hidden)

# "/osc/schema"

SSC schema reflection.

- limits (hidden)

# "/osc/version"

SSC protocol version.

- limits (hidden)

# "/osc/xid"

SSC transaction ID.

- limits (hidden)

# "/osc/ping"

SSC Ping.

- limits (hidden)

# "/osc/error"

SSC error state.

- limits (hidden)

# "/bays/update/progress"

Type: number (ro), Value range: 0..100 `[%]`, Returns always a valid value, also if device is not in update mode.

- limits
    - type: Number
    - const: false
    - writeable: false
    - count: 2
    - units: %
    - max: 100
    - min: 0
    - inc: 1
    - subscr: true

# "/bays/update/error"

Type: string (ro), Value range: “NONE” or a human readable error message

- limits
    - type: String
    - const: false
    - writeable: false
    - count: 2
    - subscr: true

# "/bays/update/enable"

Type: boolean (rw)

- limits
    - type: Boolean
    - const: false
    - writeable: true
    - count: 2
    - subscr: true

# "/bays/warnings"

More detailed description of errors in case of a non-working CHG 70N-C.

- type: Read only
- value: string example: `{"bays":{"warnings":["BatteryNotChargeable","BatteryComError"]}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 2
    - 
    - options:
        - ▷ BatteryComError
        - ▷ BatteryNotChargeable
        - ▷ BatteryNotDischargeable
        - ▷ OvercurrentDetected
        - ▷ BatteryTempOutOfRange
    - subscr: true

# "/bays/version"

Reads tx version number being plugged in CHG 70N-C.

- type: Read only
- value: string example: `{"bays":{"version":"1.6.22",""}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 2
    - options: AnyString
    - subscr: true

# "/bays/sync_settings"

Writes sync settings for inserted SKx transceivers.

- type: Write only
- value: variant example: `{"bays":{"sync_settings":[{"bay_id":0,"cable_emulation":"type1","name":"Lead Singer"},{"bay_id":1,"frequency":975525","lock":false}]}}`
- limits
    - type: String
    - const: false
    - writeable: true
    - count: -1
    - options:
        - ▷ bay_id
        - ▷ cable_emulation
        - ▷ frequency
        - ▷ led
        - ▷ link_density_mode
        - ▷ lock
        - ▷ lowcut
        - ▷ name
        - ▷ mute_config
        - ▷ trim

# "/bays/sync_error"

Reads sync error for inserted SKx transceivers.

- type: Read only
- value: string example: `{"bays":{"sync_error":["SyncFailed",""]}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 2
    - options: SyncFailed
    - subscr: true

# "/bays/state"

Reads status of a single bay of CHG 70N-C.

- type: Read only
- value: string example: `{"bays":{"state":["NORMAL","ERROR"]}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 2
    - options:
        - ▷ NORMAL
        - ▷ UPDATE
        - ▷ ERROR
        - ▷ DFU_MODE
    - subscr: true

# "/bays/serial"

Reads tx serial number being plugged in CHG 70N-C.

- type: Read only
- value: string example: `{"bays":{"serial":["123456",""]}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 2
    - options: AnyString
    - subscr: true

# "/bays/identify"

Starts flashing leds of CHG 70N-C.

- type: Read/Write
- value: bool example: `{"bays":{"identify":[true,true]}}`
- limits
    - type: Boolean
    - const: false
    - writeable: true
    - count: 2
    - subscr: true

# "/bays/device_type"

Reads type of device being plugged in CHG 70N-C.

- type: Read only
- value: string example: `{"bays":{"device_type":["BA70","EW-DX SK"]}}`
- limits
    - type: String
    - const: false
    - writeable: false
    - count: 2
    - options:
        - ▷ EW-DX SK
        - ▷ EW-DX SK 3-pin
        - ▷ EW-DX SKM
        - ▷ EW-DX SKM-S
        - ▷ BA70
        - ▷ NONE
        - ▷ UNKNOWN
    - subscr: true

# "/bays/bat_timetofull"

Reads how many minutes are needed till battery is fully loaded.

- type: Read only
- value: Integer example: `{"bays":{"bat_timetofull":null}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - count: 2
    - units: minutes
    - max: 1000
    - min: 0
    - inc: 1
    - subscr: true

# "/bays/bat_health"

Reads current state of health in % from battery.

- type: Read only
- value: Integer example: `{"bays":{"bat_health":null}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - count: 2
    - units: %
    - max: 100
    - min: 0
    - inc: 1
    - subscr: true

# "/bays/bat_gauge"

Reads current state of charge value in % from battery.

- type: Read only
- value: Integer example: `{"bays":{"gauge":null}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - count: 2
    - units: %
    - max: 100
    - min: 0
    - inc: 1
    - subscr: true

# "/bays/bat_cycles"

Reads current state of cycles from battery.

- type: Read only
- value: Integer example: `{"bays":{"bat_cycles":null}}`
- limits
    - type: Number
    - const: false
    - writeable: false
    - count: 2
    - max: 10000000
    - min: 0
    - subscr: true

# 13. SSC String characters

- ASCII 32...126 including escaped characters ASCII 34('"'), ASCII 47('/'), ASCII 92('\')
- Escaped primitives '\b', '\f', '\r' ,'\n', '\t'
- No unicode patterns '\uxxxx'

# 14. SSC Error List (CHG 70N-C)

- 100 : continue
- 102 : processing
- 200 : OK
- 201 : created
- 202 : adapted
- 210 : partial success
- 310 : subscription terminates
- 400 : message not understood
- 401 : authorisation needed
- 403 : forbidden
- 404 : address not found
- 406 : not acceptable
- 408 : request time out
- 409 : conflict
- 410 : gone
- 413 : request too long
- 414 : request too complex
- 416 : requested range not satisfiable
- 422 : unprocessable entity
- 423 : locked
- 424 : failed dependency
- 450 : answer too long
- 454 : parameter address not found
- 500 : internal server error
- 501 : not implemented
- 503 : service unavailable