trivum API documentation

trivum API

trivum technologies GmbH <info@trivum.com>

v0.1, 2024-12-13 :title-logo-image: image::../images/trivum-logo.svg[pdfwidth=150,align=right]

trivum HTTP Interface

25-Jul-2023

The trivum HTTP interface takes requests which can be tested easily by a web browser, and returns replies in XML format.

1. Commands

1.1. ZoneCommand

Allows to do basic things like switching a zone off, or to change volume.

```
/xml/zone/runCommand.xml?zone=@zoneId&command=commandNumber
```

zoneId

The ID of a zone. For the list of possible IDs look into the web configuration under `Automation /trivum API` or see below the getAll.xml example.

Some Actuators may not address the first zone by @0 but by @1 due to internal, unused configuration file(s). To fix this, you may reset the whole configuration by: System / Backup/Restore / Reset all configuration data

Instead of `@0` the zone name can be given. If it contains special characters rewrite them using `%` :

```
/xml/zone/runCommand.xml?zone=living%20room&command=...
```

commandNumber

This is a numeric command with these possible values:

```
ZONECMD_POWER_OFF            001
ZONECMD_MUTE                 002   toggle mute
ZONECMD_MUTE_ON              680   since v9.29
ZONECMD_MUTE_OFF             681   since v9.29
ZONECMD_VOLUME_INC           003
ZONECMD_VOLUME_DEC           004
ZONECMD_VOLUME_INC2          009
ZONECMD_VOLUME_DEC2          010
ZONECMD_VOLUME_INC5          011
ZONECMD_VOLUME_DEC5          012
ZONECMD_ALLOFF               015
ZONECMD_SNOOZE               017
ZONECMD_LOCAL_SOURCE         019   if present (LineIn)
ZONECMD_USE_PREV_SOURCE      029   see Zones / zone / KNX/HTTP sources
ZONECMD_JOIN                 030
ZONECMD_UNJOIN               031
ZONECMD_USE_NEXT_SOURCE      041   see Zones / zone / KNX/HTTP sources
ZONECMD_USE_NEXT_ZONE        042   see Zones / zone / KNX/HTTP sources
ZONECMD_DEFAULT_STREAMING    050   if present
ZONECMD_DEFAULT_TUNER        051   if present
ZONECMD_VOLUME_DEC_1         080
ZONECMD_VOLUME_DEC_10        089
ZONECMD_VOLUME_INC_1         090
ZONECMD_VOLUME_INC_10        099
MULTIKEY_BASIC_FORWARD       400   skip to next  track, preset
MULTIKEY_BASIC_BACKWARD      401   skip to prev. track, preset
MULTIKEY_BASIC_FASTFORWARD   402
MULTIKEY_BASIC_FASTBACKWARD  403
```

```
MULTIKEY_BASIC_PLAYPAUSE      406
MULTIKEY_PLAY                 431
MULTIKEY_PAUSE                432
MULTIKEY_STOP                 433
MULTIKEY_STATION_DOWN         490
MULTIKEY_STATION_UP           491
MULTIKEY_NEXT_ALBUM           493
MULTIKEY_PREVIOUS_ALBUM       494
MULTIKEY_NEXT_PLAYLIST        495
MULTIKEY_PREVIOUS_PLAYLIST    496
ZONECMD_START_PAGING_1        500
ZONECMD_START_PAGING_32       531
ZONECMD_STOP_PAGING_1         550
ZONECMD_STOP_PAGING_32        581
ZONECMD_STOP_PAGING_ALL       599
ZONECMD_PRESET_1              600
ZONECMD_PRESET_7              606
ZONECMD_GROUP_START_1         621
ZONECMD_GROUP_START_8         628
ZONECMD_GROUP_STOP            630
ZONECMD_GROUP_STOP_1          631
ZONECMD_GROUP_STOP_8          638
ZONECMD_GROUP_STOP_ALL        639
ZONECMD_STREAMING_NOPLAY      641
ZONECMD_VOLUME_00             900
ZONECMD_VOLUME_99             999
ZONECMD_ROOM_VOLUME_00       1000
ZONECMD_ROOM_VOLUME_99       1099
```

Examples

call	function
`/xml/zone/getAll.xml`	List all possible zone ID's.
`/xml/zone/get.xml?zone=@0`	Get status of a single zone. Optional parameters are: &addSourceBasicData &addSourceStatusData
`/xml/zone/getSelection.xml?grouped`	List zones with full group informations.
`/xml/zone/runCommand.xml?zone=@0&command=50`	Switch the first zone to default streaming.
`/xml/zone/runCommand.xml?zone=@0&command=1`	Switch the first zone off.
`/xml/zone/runCommand.xml?zone=@0&command=15`	Switch all zones off.
`/xml/zone/runCommand.xml?zone=@0&command=680`	Mute on
`/xml/zone/runCommand.xml?zone=@0&command=681`	Mute off

1.2. Set Zone Source

Select a zone source by short name

```
/xml/zone/set.xml?zone=@0&source=@shortSourceName
```

shortSourceName

text	action	remark
a a1 a3	first analog input first analog input third analog input	Depending on the device model, 0 to 8 analog inputs are supported.
p p5	first FM tuner preset fifth FM tuner preset	Requires that a default FM tuner is configured for the zone.
f f2	first trivum favorite second trivum favorite	
y y2	first trivum playlist second trivum playlist	
i i2	first tunein preset second tunein preset	
s	default stream source of zone	playing recent selection
t	default FM tuner of zone	playing recent frequency

Examples

API call	remark
/xml/zone/set.xml?zone=@0&source=@a1	switch to first analogue input
/xml/zone/set.xml?zone=@0&source=@t	switch to default FM tuner of zone and play recent frequency
/xml/zone/set.xml?zone=@0&source=@p3	switch to default FM tuner of zone and play station preset 3
/xml/zone/set.xml?zone=@0&source=@f2	switch to default streaming of zone and play trivum favorite 2
/xml/zone/set.xml?zone=@0&source=@i5	switch to default streaming of zone and play TuneIn webradio preset 5
/xml/zone/set.xml?zone=@1@source=@n	C4 only: use a source by card slot n. (n >= 0)
/xml/zone/runCommand.xml?zone=@0&command=15	Switch all zones off.

1.3. Set Zone Attribute

Change basic values in a zone, like volume, mute, balance or bass.

API call	remark
/xml/zone/set.xml?zone=@0&volume=10	set volume (0 ... 100)

`/xml/zone/set.xml?zone=@0&action=1`	same as `/xml/zone/runCommand.xml` to run a numeric command, in this case `ZONECMD_POWER_OFF (1)`
`/xml/zone/set.xml?zone=@0&balance=0`	set balance, from -15 (full left) to 15 (full right)
`/xml/zone/set.xml?zone=@0&bass=-5`	set bass reduction or enhancement, from -15 to 15
`/xml/zone/set.xml?zone=@0&treble=5`	set treble reduction or enhancement, from -15 to 15

1.4. trivum Favorites

To create trivum favorites:

- play some music content, like a NAS album
- then select ⋯ at the right top
- then select "Add to trivum favorites".

Get the list of trivum favorites:

`/api/v1/trivum/favorite.xml`

Play a trivum favorite:

`/xml/zone/set.xml?source=@f1&zone=@0`

You may also add options:

option	remark
`&sequence=random-sequential`	select a random start track
`&sequence=random-random`	play in permanent random order

1.5. trivum Playlists

Get the list of trivum playlists:

`/api/v1/trivum/playlist.xml`

Play a trivum playlist:

`/xml/zone/set.xml?source=@y1&zone=@0`

You may also add options:

option	remark
`&sequence=random-sequential`	to start at a random track
`&sequence=random-random`	to play only random tracks

1.6. TuneIn Favorites

These can also be created by ⋯ at the right top while a TuneIn station is playing.

Get the list of TuneIn favorites:

`/api/v1/tunein/favorite.xml`

Play a TuneIn favorite:

`/xml/zone/set.xml?source=@i1&zone=@0`

1.7. FM presets

List FM presets:

`/xml/system/getTunerStationList.xml`

On C4 this shows the system wide list of FM presets, but no local presets stored per FM tuner card.

1.8. NAS status and control

API call	remark
`/xml/system/getMusicCenterStatus.xml`	get NAS library status
`/xml/system/scanMusicCenterShares.xml`	rerun full NAS scan

1.9. Group management

Groups can be created, changed or removed by one call:

```
/xml/zone/createGroup.xml?zone=zVisu&oldgroup=zMaster&members=++----------
```

Parameters:

name	remark
zVisu	index of the current zone of the visualization client
zMaster	index of the group master whose music should be used (if both zones are currently playing different sources)
+/-	characters telling graphically which zones should take part in a group. for example, with a 4 zone system, type 4 characters or less (is filled up with - automatically).

Example: second zone joins playback of first zone

- first zone is playing a stream, second zone is playing FM tuner, all other zones are off.

- second zone should be added to a group with first zone,
  and it should take over music from first zone (the stream).

`/xml/zone/createGroup.xml?zone=1&oldgroup=0&members=++--`

Result: the second zone starts playing the same stream as first zone.

Example: first zone joins playback of second zone

- first zone is playing a stream, second zone is playing FM tuner, all other zones are off.
- first zone should be added to a group with second zone, and it should take over music from second zone (the tuner).

`/xml/zone/createGroup.xml?zone=0&oldgroup=1&members=++--`

Result: first zone starts playing the same FM tuner as second zone.

This means if both zones are playing different sources then
"oldgroup" decides which music to play after the group join.

Example: second zone should leave the group

`/xml/zone/createGroup.xml?zone=0&oldgroup=0&members=+---`

Relevant here is the change from `+` to `-` in the members list.

Change volume level within a group

Within a group, zones normally do not use isolated volume levels,
but a change in volume affects all group members.
This interdependency is handled by the call:

`/xml/zone/setVolume.xml`

By default, this call will not simply **set** an absolute volume level, but it **steps a bit** into the direction of a given target volume. This is best used with a + or - button in your visualization.

API call	remark
`/xml/zone/setVolume.xml?id=@0&volume=0`	Step the group volume downwards for the whole group. id is any zone ID from the group. The volume of all zone members will be decreased a few steps.
`/xml/zone/setVolume.xml?id=@0&volume=99`	Step the group volume upwards for the whole group. The volume of all zone members will be increased a few steps.
`/xml/zone/setVolume.xml?id=@0&groupMemberVolume=99`	Increase volume of a single zone step wise, not affecting other group members.
`/xml/zone/setVolume.xml?id=@0&groupMemberVolume=0`	Decrease volume of a single zone step wise, not affecting other group members.

`/xml/zone/setVolume.xml?id=@0&stop`	Stop volume stepping immediately.
`/xml/zone/setVolume.xml?id=@0&groupMemberVolume=50&absolute`	Set **absolute** volume for a single zone, isolated from other group members. (Use with care.)

To get the new volume level informations within a group make a getChanges call and look into the volume status list.

`/xml/zone/getChanges.xml?zone=@0&visuid=90&apiLevel=2&now`

Example output, if grouped, under zone / status:

```
<zone>
 ...
 <status>
 <volume>17</volume>  - volume of zone making the getChanges call
 ...
 <group>
  <zone>0</zone>
  <volume>17</volume> - volume for zone id 0
 </group>
 <group>
  <zone>1</zone>
  <volume>26</volume> - volume for zone id 1
 </group>
 <group>
   ...
 </group>
 <groupMembers>2</groupMembers>
 </status>
</zone>
```

For a full explanation of getChanges see Get Zone Status.

1.10. Paging

Pagings must be configured in the web configuration. Then the following calls can be used:

Start paging

`/xml/paging/start.xml`

Parameters

name	description
`id`	paging ID, 0 - 31
`volume`	optional, 5 - 100. if not supplied the configured paging volume level is used.

name	description
autostoptime	optional, 5 - 100 seconds. if not supplied the configured stop settings are used.

Example

/xml/paging/start.xml?id=0&volume=10&autostoptime=10

A paging stops automatically after the defined time, but you may stop it earlier by calling:

/xml/paging/stop.xml?id=0

2. Interactive Music Selection

Starts with:

/xml/system/getWebTouchMenu.xml?which=music&zone=@0&visuid=90

This produces records like:

```
<row>
  <type>action</type>
  <mode>menu</mode>
  <action>/xml/system/getWebTouchMenu.xml?which=trivumFavorites&amp;keypad=4</action>
  <icon>/imgs/visuIconServiceFavorites_128px.png</icon>
  <text>trivum_20favorites</text>
</row>
```

then, per record:

- decode and display the text field in your visualization.
  _20 means a character with Ascii Code 0x20 (a space).
- if touched, call the action url and display the next menu level.

Do not rely on the permanent availability of specific menu levels.
Especially the menus provided by music services may change over time.

3. Get Zone Status

3.1. Synchroneous

Poll the status of a zone with one short API call:

/xml/zone/getChanges.xml?zone=@0&visuid=90&apiLevel=2&now

Parameters

name	function
`visuid`	a number from 1 to 99 to identify your external visualization instance. within this API document, visuid=90 is used for test requests.
`apiLevel`	should always be 2. this will produce `button` xml objects under `keypad / basic`.
`now`	tells the server to return the new zone status immediately, and to close the connection. without `&now` the call would block until a timeout, or until a change in the zone status informations.
`reload=1`	if two visualizations access the same server with the same visuid an error "used twice" may appear. in this case the most recent visu should add `&reload=1` on the first call, to tell clearly it is the most recent visualization.

About Control units (Visualizations)

If you send requests with `visuid=90` a *Control Unit* object with ID 90 is created in the server.

You can get the list of current Control Units in the web configuration, under *Control Units*.

After first access, the unit is listed as "Not configured". As soon as you change it's configuration, e.g. by setting the option "Off by a short press on power", it is called *Configured*, and later cleanups of the Control Units list will not delete this one.

If there are no requests for this unit, after some time it will be listed under "currently inactive control units".

3.2. Asynchroneous

This means a HTTP call will not return immediately, but it will block until something changes.

Example:

`/xml/zone/getChanges.xml?zone=@0&visuid=90&apiLevel=2`

Notice that `&now` is missing. The following will happen:

on first API call:

A Control Unit with ID 90 is created, and linked with the first zone.
The API call returns immediately, with full status data of the zone.

on all further API calls:

The existing Control Unit 90 is reused. The API call may block, until:

- a timeout is reached (10 seconds approx.). in this case you get a reply like:
  `<rows><system><timeout>1</timeout>`
- or until something changed, for example, the volume in the zone.

if (many) status data has changed at the server between two getChanges calls, the call may not block at all, but return the new status immediately.

when you receive a timeout, just re-run the getChanges immediately. this means you may run getChanges endlessly, in a loop, for example in a separate I/O thread. Because a request returns only on changes, this will cause no load problems with the server.

when you do not receive a timeout, i.e. the call returns immediately or after a few seconds (as soon as something has changed), then process the status data, and then re-run the getChanges request.

3.3. Appendix: schematic example for a Visu Client Application

3.3.1. Single Thread Application

This requires that you can test, in your programming language, if reply data for a socket exists (via select() call).

Main Thread

- start: send `/xml/zone/getChanges.xml?visuid=90&now`
- loop begin: update the GUI.
  - process input events from user.
  - send synchroneous commands like:
    `/xml/zone/runCommand.xml?…`
    receive reply, check rc AND process xml status data
    (same as with getChanges replies)
  - check if reply data exists for ongoing getChanges call
    (in C code: select() call on socket)
    IF data exists from trivum server:
    - Look for `<userdata name="rc">0</userdata>`.
      If NOT present
      _ process the error and wait a few seconds.
      Else if NOT a timeout
      _ process xml reply (status data)
      Endif
      async call (just send)
      `/xml/zone/getChanges.xml&visuid=90&onlyChanges`
      Endif

    - if no data from server arrive within 1 minute
        - async call (just send)
          `/xml/zone/getChanges.xml&visuid=90&onlyChanges`
          endif
    - re-run loop

3.3.2. Two Thread Application Example

Can be used if you prefer to run blocking receives on sockets in a separate I/O thread.

Main Thread

- update the GUI.
- process input events from user.
- send synchroneous commands like:
  `/xml/zone/runCommand.xml?…`
  receive reply, check rc AND process xml status data
  (same as with getChanges replies)
- receive status data and errors from Status Thread.
- re-run this loop.

Status Thread

- IF on first loop:
    - send `/xml/zone/getChanges.xml?visuid=90&now`
      ELSE
    - send `/xml/zone/getChanges.xml?visuid=90&onlyChanges`
- receive reply (this is blocked up to 10 seconds)
- Look for `<userdata name="rc">0</userdata>`.
  If this is NOT present then there is an error.
  Make sure not simply to re-run the loop on errors,
  but at least wait a few seconds, and tell the Main Thread.
- Look for `<rows><system><timeout>1</timeout>`.
  IF this is present
    - re-run the loop immediately.
      ELSE
    - process the reply status data,
      and copy new status data to Main Thread.
- re-run this loop.