Neovid REST HTTP API

The Neovid app supports the REST interface for information management.

Storage Service

Receiving Media Asset List

A request to the media database for retrieving the list of media assets. The request returns an XML-document with short descriptions of media assets in the whole database or in a certain folder. Graphic templates have the mime_type field with the value “10”.

```
GET
http://<IP_ADDR>:7901/storage/<NAME>[/<FOLDER>[/<FOLDER>...]][?format=val[&datetime=val...]]
```

- **GET** – GET request by the HTTP protocol;
- **<IP_ADDR>** – the server IP address;
- **<NAME>** – the media database name;
- **<FOLDER>** – the folder name;
- **format** – the unnecessary “format” parameter defines formats of XML output:
  - **StoredTree** - the XML tree of folders and assets in the media database (default),
  - **mam** – linear asset output,
  - **folderlist** – retrieving the list of folders. Added in the 2.3.90 version.
- **datetime** – the unnecessary “datetime” parameter filters output according to the asset date of creation. When specifying a value in the “ISO date time” format, the server will output assets created later than the specified date.

An example of server response to a request to the media database in the StoredTree mode:

| response_storedtree_mediadb.xml

```
<StoredTree query_root="">
    <Folder name="">
        <Clip link_id="0" finalized="true">
            <Attr name="bps">16</Attr>
            <Attr name="channels">2</Attr>
            <Attr name="derived_from">CityViews2 HD</Attr>
            <Attr name="frame_time">0.0400000000</Attr>
            <Attr name="frames">3425</Attr>
            <Attr name="height">1080</Attr>
            <Attr name="orig_name">CityViews2 HD</Attr>
            <Attr name="orig_type">3</Attr>
            <Attr name="sample_rate">48000</Attr>
            <Attr name="width">1920</Attr>
            <Attr name="first_frame">0</Attr>
            <Attr name="last_frame">3425</Attr>
```

```
            <Attr name="title">CityViews3</Attr>
            <Attr name="create_time">2018-03-27T10:26:30</Attr>
            <Attr name="rec_time">2016-12-17T16:15:14</Attr>
        </Clip>
        <Clip link_id="3" finalized="true">
            <Attr name="bps">16</Attr>
            <Attr name="channels">2</Attr>
            <Attr name="derived_from">Hockey</Attr>
            <Attr name="frame_time">0.0400000000</Attr>
            <Attr name="frames">1501</Attr>
            <Attr name="height">1080</Attr>
            <Attr name="orig_name">Hockey</Attr>
            <Attr name="orig_type">3</Attr>
            <Attr name="sample_rate">48000</Attr>
            <Attr name="width">1920</Attr>
            <Attr name="first_frame">0</Attr>
            <Attr name="last_frame">1501</Attr>
            <Attr name="title">Hockey</Attr>
            <Attr name="create_time">2018-03-27T14:13:15</Attr>
            <Attr name="rec_time">2016-12-18T01:14:24</Attr>
        </Clip>
        <Folder name="New Folder"/>
        <Folder name="New Folder 1"/>
    </Folder>
</StoredTree>
</xml>
```

An example of server response to a request to the media database in the mam mode:

| response_mam_mediadb.xml

```
<mam>
    <asset>
        <id>CityViews3</id>
        <dir/>
        <length>00:02:17</length>
        <hires>
        $(video_path)/media/CityViews2_HD-NID0-3/CityViews2_HD.avi
        </hires>
        <proxy>
        $(video_path)/proxy/CityViews2_HD-NID0-19/CityViews2_HD_lrv.avi
        </proxy>
    </asset>
    <asset>
        <id>Hockey</id>
        <dir/>
        <length>00:01:00.04</length>
        <hires>$(video_path)/media/Hockey-NID3-51/Hockey.avi</hires>
    </asset>
```

```
</mam>
```

An example of server response to a request to the media database in the folderlist mode:

| response_folderlist_mediadb.xml

```
<StoredTree query_root="">
    <Folder name="">
        <Folder name="New Folder"/>
        <Folder name="New Folder 1"/>
    </Folder>
</StoredTree>
```

Retrieving Information on Asset

A request for receiving detailed information on the object (including names of parameters for graphic templates). The request returns an XML-document.

```
GET http://<IP_ADDR>:7901/storage/<NAME>@<TITLE>
```

An example of server response to a request to the media database for retrieving detailed information on an object:

| response_asset_mediadb.xml

```
<Clip folder="" hires="$(video_path)/media/CityViews2_HD-
NID0-3/CityViews2_HD.avi" proxy="$(video_path)/proxy/CityViews2_HD-
NID0-19/CityViews2_HD_lrv.avi" archive="" link_id="0" finalized="true">
    <Attr name="bps">16</Attr>
    <Attr name="channels">2</Attr>
    <Attr name="derived_from">CityViews2 HD</Attr>
    <Attr name="frame_time">0.0400000000</Attr>
    <Attr name="frames">3425</Attr>
    <Attr name="height">1080</Attr>
    <Attr name="orig_name">CityViews2 HD</Attr>
    <Attr name="orig_type">3</Attr>
    <Attr name="sample_rate">48000</Attr>
    <Attr name="width">1920</Attr>
    <Attr name="first_frame">0</Attr>
    <Attr name="last_frame">3425</Attr>
    <Attr name="title">CityViews3</Attr>
    <Attr name="create_time">2018-03-27T10:26:30</Attr>
    <Attr name="rec_time">2016-12-17T16:15:14</Attr>
</Clip>
```

Receiving WebProxy

The request returns media data (MP4 or HLS).

```
GET http://<IP_ADDR>:7901/storage/<NAME>@<TITLE>/<WPN>
```

- **GET** – a GET request by the HTTP protocol;
- **<IP_ADDR>** – the server IP address;
- **<NAME>** – the media database name;
- **<TITLE>** – the asset (clip) name;
- **<WPN>** – the web-profile name. Possible values: WP1-WP8. If a requested profile is not configured in media database settings, you will receive an error message `HTTP ERROR 500`. If a WebProxy-copy with a specified profile has not been created earlier for this asset, it will be generated in real time. This process may take some time.

Receiving clip XMP metadata

```
GET http://<IP_ADDR>:7901/storage/<NAME>@<TITLE>/GETUSERFIELDS
```

Asset Deletion

```
DELETE http://<IP_ADDR>:7901/storage/<NAME>@<TITLE>
```

Loading XMP Metadata

The detailed description of the function of XMP metadata batch load to MAM through Metadata Connectors. Added in the 2.0.60 version.

```
PUT http://<IP_ADDR>:7901/storage/<NAME>?connector=<CONNECTOR_NAME>
```

Actions with Assets

```
POST
http://<IP_ADDR>:7901/storage/<NAME>@<TITLE>?command=<COMMAND_NAME>[&param=val...]
```

- **POST** – a POST request by the HTTP protocol;
- **<IP_ADDR>** – the server IP address;
- **<NAME>** – the media database name;
- **<TITLE>** – the asset (clip) name;
- **<COMMAND_NAME>** – the executable command;
  - **RESTORE** – restoring HIRES from an archive copy. Added in 2.0.160 version.
  - **COPYLINK** – creating a new link to this asset;
    - param1 – **title** – title of a new link;
    - param2 – **folder** – the folder where a new link will be created;

- **SETFIELD** – change the link field value;
  - param1 – **name** – the field name («type», «clip_name», «comment», «layer», vfirst_frame», «last_frame», «icon_pos», «user_1» ... «user_8»)
  - param2 – **val** – a new field value;
- **SETUSERFIELDS** - change the values of XMP fields. Added in version 2.6.1.
  - param1 is the parameter name specifies the name of the XML field whose value is to be changed, val is a comma-separated list of new values.
  - ...
  - paramN
- **CREATESEGMENT** - add segment marker to asset metadata
  - param1 - **seg_num** - the number of the segment to create
  - param2 - **start_frame** - segment start (timecode in form HH: MM: SS [:;] FF or HH: MM: SS.ss or frame number)
  - param3 - **duration** - segment duration (timecode in form HH: MM: SS [:;] FF or HH: MM: SS.ss or frame number)
  - param4 - **title** - segment name (optional)
  - param5 - **keywords** - keywords for search (optional)
- **DELETESEGMENT** - removes the segment marker from the asset metadata.
  - param1 - **seg_num** - segment number to be deleted.
- **CREATEMARKER** - Adds or modifies a keyframe marker in the asset's metadata. Added in version 2.6.1.
  - param1 - **uid** - a unique integer marker identifier that will be used to modify an existing marker. A value of -1 will generate a new identifier when creating a token.
  - param2 - **frame** - marker position (in frames)
  - param3 - **track** - marker type, allowed values:
    - «General»,
    - «Secondary Event»,
    - «Cue Point»,
    - «Goto Cue»,
    - «Game Time»,
    - «Icon»,
    - «Picture»,
    - «Profanity Start»,
    - «Profanity End»
  - ...
  - paramN - Name / value fields can be set for the marker by passing them as a “name = value” pair. The default fields that are available in markers are:
    - «Title»,
    - «Comment»,
    - «User»,
    - «Keywords»
- **DELETEMARKER** - Removes the keyframe marker from the asset's metadata. Added in version 2.6.1.
  - param1 - **uid** - unique integer identifier of the marker to be removed.

Generation of PNG Thumbnail for Asset

A request for retrieving a PNG thumbnail of the media asset (including parameterized graphic templates) for any specified frame. The request returns a PNG image. Added in the 2.3.63 version.

```
GET
http://<IP_ADDR>:7901/storage/<NAME>@<TITLE>/GETICON[?[pos=FRM][&dur=FRM][&w
=PX][&h=PX][&param=NAME:VAL[&param=NAME:VAL]]]
```

- **GET** – a GET request by the HTTP protocol;
- **<IP_ADDR>** – the server IP address;
- **<NAME>** – the media database name;
- **<TITLE>** – the asset (clip) name;
- **pos=FRM** – the thumbnail position in frames from the asset start (the IN point);
- **dur=FRM** – duration of the graphic composition in frames; This option is used in case, when a composition has infinite duration, and different animation effects depend on actual duration of the composition.
- **w=PX** – width of the retrieved thumbnail in pixels (the default value equals width of the initial image);
- **h=PX** – height of the retrieved thumbnail in pixels (the default value equals height of the initial image);
- **param=NAME:VAL** – parameters sent to the graphic composition while rendering:
  - **NAME** – the graphic element name (for instance: Text Area 0);
  - **VAL** – the value sent to the graphic element with the NAME title.

RSS Feeder Service

```
GET http://<IP_ADDR>:7901/rss/<NAME>
```

```
PUT http://<IP_ADDR>:7901/rss/<NAME>/<FNAME>
```

```
DELETE http://<IP_ADDR>:7901/rss/<NAME>?guid=<GUID>
```

GPI Service

A request for retrieving the latest GPI event. The request returns an XML-document.

```
GET http://<IP_ADDR>:7901/gpi?module_id=<N>
```

Parameters description:

- **GET** – a GET request by the HTTP protocol;
- **<IP_ADDR>** – the server IP address;
- **module_id** – the GPI_Board service sequence number (starts with 1).

An example of a request to the server:

```
http://localhost:7901/gpi?module_id=2
```

An example of the server response for the GPI Board System/HotKey service:

| response_gpi.xml

```
<GPI ReceivedTime="2018-03-28T14:00:42.904">
    <Attr name="last_key">Alt+F1</Attr>
</GPI>
```

Program Channel

A request for retrieving the latest GPI event. The request returns an XML-document.

```
POST http://<IP_ADDR>:7901/pgm?channel=<N>&command=<cmd>
```

Parameters description:

- **GET** – a GET request by the HTTP protocol;
- **<IP_ADDR>** – the server IP address;
- **<N>** – the ordinal number of the Program service (starts with 1);
- **<COMMAND_NAME>** – the executable command;
    - **scte104_timesignal** - Generate SCTE-104 “TimeSignal” message. The body of the request should contain XML description of the message to be generated.

AsRun

A request for retrieving a report on executed broadcast events (the log AsRun). The request returns an XML-document.

```
GET
http://<IP_ADDR>:7901/as_run?module_id=<N>[&max_count=M][&datetime=YYYY-MM-D
DTHH:MM:SS]
```

Parameters description:

- **GET** – a GET request by the HTTP protocol;
- **<IP_ADDR>** – the server IP address;
- **module_id** – the sequence number of the program channel service (starts with 1);
- **max_count** – the filter by the number of recordings displayed from the end of the list (show the last M recordings);
- **datetime** – the filter by the time of playout start (display recordings with the “start_time” value more than specified in this field). The time is specified in the YYYY-MM-DDTHH:MM:SS format.

An example of a request to the server:

```
http://localhost:7901/as_run?module_id=1&max_count=3&datetime=2018-03-28T13:
29:00
```

An example of server response:

| response_asrun.xml

```
<AsRun request_time="2018-03-28T13:30:38.839">
    <BroadCasted_Clip uri="CityViews2 HD">
        <Attr name="title">CityViews2 HD</Attr>
        <Attr name="start_time">2018-03-28T13:29:04.560</Attr>
        <Attr name="end_time">2018-03-28T13:29:57.160</Attr>
        <Attr name="duration">00:00:52:15</Attr>
        <Attr name="onair"/>
    </BroadCasted_Clip>
    <BroadCasted_Clip uri="Robots 1">
        <Attr name="title">Robots 1</Attr>
        <Attr name="start_time">2018-03-28T13:29:57.160</Attr>
        <Attr name="end_time">2018-03-28T13:30:24.160</Attr>
        <Attr name="duration">00:00:27:00</Attr>
        <Attr name="onair"/>
    </BroadCasted_Clip>
    <BroadCasted_Clip uri="Robots 2">
        <Attr name="title">Robots 2</Attr>
        <Attr name="start_time">2018-03-28T13:30:24.160</Attr>
        <Attr name="end_time">2018-03-28T13:30:36.120</Attr>
        <Attr name="duration">00:00:11:24</Attr>
        <Attr name="onair"/>
    </BroadCasted_Clip>
</AsRun>
```

Recorder service

Requests to control the recording service. Added in version 2.5.61.

Status request

```
GET http: // <IP_ADDR>: 7901 / recorder? Module_id = <N>
```

Parameters description:

- **GET** – a GET request by the HTTP protocol,
- **<IP_ADDR>** – the server IP address,
- **<N>** – the ordinal number of the Recorder service (starts with 1).

Selecting a recording profile

```
POST http: // <IP_ADDR>: 7901 / recorder? Module_id = <N> & command =
set_profile & profile = <NAME>
```

Parameters description:

- **POST** – POST request via HTTP protocol,
- **<IP_ADDR>** – the server IP address,
- **<N>** – the ordinal number of the Recorder service (starts with 1),
- **<NAME>** – the name of the Destination Profile. A profile with a specified name must be preconfigured in the recording service.

Start recording

```
POST http: // <IP_ADDR>: 7901 / recorder? Module_id = <N> & command =
rec_start [& title = <TITLE>] [& folder = <FOLDER> |.]
```

Parameters description:

- **POST** – POST request via HTTP protocol,
- **<IP_ADDR>** – the server IP address,
- **<N>** – the ordinal number of the Recorder service (starts with 1),
- **<TITLE>** – the name of the recorded clip,
- **<FOLDER>** – is the name of the target subfolder that will be used to save the record.

Stop recording

```
POST http: // <IP_ADDR>: 7901 / recorder? Module_id = <N> & command =
rec_stop
```

Parameters description:

- **POST** – POST request via HTTP protocol,
- **<IP_ADDR>** – the server IP address,
- **<N>** – the ordinal number of the Recorder service (starts with 1),

Returned XML fields:

- **Result** – operation result:
  - **SUCCESS** - Operation successful
  - **ERR_JOB_FAILED** - Recording job creation failed
  - **ERR_NO_PRF** - No such recording profile
  - **ERR_REC_ACTIVE** - Can't change recording profile during active recording job
  - **ERR_INTERNAL** - Internal error
  - **ERR_UNK_CMD** - Unknown command
  - **ERR_COMM_FAILED** - Recorder server offline or not responding
  - **ERR_NO_CMD** - No command is given in the request
- **JobId** - current / created job id
- **Title** - job title / filename base
- **State** - job state
  - **INVALID**
  - **INVACTIVE**
  - **CUING**
  - **CUED**
  - **RECORDING**

    - **STOPPED**
    - **PAUSED**
- **Folder** - job folder
- **FramesDropped** - number of frames dropped during recording
- **StartTime** - server time when recording started
- **CurrentTime** - current server time
- **Profile** - current profile

Insert a marker to the current recording position

```
POST
http://<IP_ADDR>:7901/recorder?module_id=<N>&command=append_marker[&title=<TITLE>][&comment=<COMMENT>][&track=<TRACK>][&keywords=<KEYWORDS>][&time_pos=<LOCAL_TIME>]
```

Parameters description:

- **POST** – POST request via HTTP protocol,
- **<IP_ADDR>** – the server IP address,
- **<N>** – the ordinal number of the Recorder service (starts with 1)
- **<TITLE>** – Used for “Profanity Start”/“Profanity End”. Values are «Blur», «Beep», «Mute», «Blur+Beep», «Blur+Mute»
- **<COMMENT>** – Optional description field
- **<TRACK>** – The name of the track. Track value names are «General», «Event», «Secondary Event», «Icon», «Picture», «Profanity Start», «Profanity End», «Cue Pint» and «GoTo Cue»
- **<KEYWORDS>** – Keywords field. For Profanity Start is used for the rectangle area of the frames.
- **<LOCAL_TIME>** - Local time which corresponds to the position of the marker. The time is given in HH:MM:SS.ms