NARRA

Petr Pulc
Faculty of information technology, CTU in Prague

Michal Mocňák, Kryštof Pešek
Centre of audiovisual studies, FAMU in Prague

April 28, 2015

Contents

Chapter 1

API v1

Instance of NARRA software can be controlled only through an API interface provided by the `narra` package. All other components (even the user interface) have to use resources provided by this API. Following text will provide basic information about initialization requirements as well as currently supported ways of authentication and access to resources in NARRA.

Individual resources are described in this part as follows:

`REQUEST_TYPE URN/with/<variable1>/<variable2>` (accessible by)

In case of request with variables, simple example is attached:

Example: `GET URN/with/filled/value`

In case of POST requests, parsed JSON payload example is attached as well. Required parameters in data payload are marked with exclamation mark:

Example: `POST URN`
Payload:

```
required:  "value" !
other:  "value"
```

All request descriptions are companied by example JSON response from server.

1.1 Topology and initialization

NARRA API serves as a standard connector between the NARRA core and all plugins and extensions that want to access data stored in NARRA.

We do not use local identities in NARRA, users have to authenticate using an OAuth identity. In time of writing of this document, Google and GitHub identity providers were supported. But to list all providers, please visit API resource `/auth/providers` of your instance.

However to make these providers work, Client IDs and Client Secrets have to be provided as an environment variables.

Standard naming scheme of OAuth environment variables is `{PROVIDER}_CLIENT_[ID,SECRET]`. For example to make Google OAuth work, environment variables `GOOGLE_CLIENT_ID` and `GOOGLE_CLIENT_SECRET` have to be set correctly.

Note: Please make sure, that the requested data (email address and basic profile info) are really provided by the API. Especially in case of Google, where individual Google APIs have to be turned on. In our case it is a Google Contacts API and Google+ API.

Reminder: In case of a docker container, environment variables can be passed by an `-e` parameter.

1.2 Authorization levels

Four basic levels of authorization are present in NARRA API:

public Everyone can access freely this resource, without even presenting its identity.

user Only authenticated users can access this resource.

author Only authenticated users with Author role can access this resource.

admin Only authenticated users with Admin role can access this resource.

User roles can be added only by Admin, with call of `POST v1/users/<username>/update`. See 1.4.

1.3 Authentication

To access vast majority of resources, user has to be signed in to the instance. If environment variables for selected identity provider(s) are set in accordance to previous text, whole process of authentication should be carried out in following way.

1.3.1 Selection of identity provider

User should be provided with selection of identity providers. The list of supported identity providers can be gathered on:

`GET auth/providers` (public)

Response can look like:

```
{"status":"OK","providers":["google","github"]}
```

1.3.2 Authentication request

For every provider the API has a way to initialize authentication request. If Google OAuth is supported, URL

`GET auth/google` (public)

will redirect us to authentication landing page of Google OAuth identity provider.

After successful authentication, user will be redirected to a NARRA API login callback URL (`auth/google/callback`) and presented with a token to keep authentication valid.

`GET auth/google/callback` (public)
`POST auth/google/callback` (public)

Responds with user token to access NARRA resources:

```
{"status":"OK","token":"MTRzNjc2MhAxNDYzORk0NTgwAjUv"}
```

At least first request to NARRA API has to contain copy this token in data payload. Other requests should have the token present as long as some Cookie storage is enabled.

1.3.3 Authentication cancelling

If we want to sign user out, we should navigate him to URL `v1/users/me/signout`, however this resource will be discussed in section 1.4.1.

1.4 User-related resources

As discussed before, user accounts are linked to identity providers and therefore, no registration process is involved. User accounts are constructed on first authentication form hash returned by the OAuth service.

Resource Users also provides information about user roles.

`GET v1/users` (author, admin)

Returns list of all users in system:

```
{"status":"OK","users":[
  {"username":"alice",
   "name":"Alice Example",
   "email":"alice@example.org",
   "image":"https://avatars.githubusercontent.com/u/1111111?v=3",
   "roles":["admin"],
   "identities":["github"]},
  {"username":"bob",
   "name":"Bob Example",
   "email":"bob@example.org",
   "image":"https://lh3.googleusercontent.com/.../photo.jpg?sz=50",
   "roles":["author","admin"],
   "identities":["google"]}
]}
```

`GET v1/users/<username>` (admin only)

Example: `GET v1/users/alice`

Returns information on one selected user:

```
{"status":"OK","user":{
  "username":"alice",
  "name":"Alice Example",
  "email":"alice@example.org",
  "image":"https://avatars.githubusercontent.com/u/1111111?v=3",
  "roles":["admin"],
  "identities":["github"]}
}
```

`GET v1/users/<username>/delete` (admin only)

Example: `GET v1/users/alice/delete`

Deletes the user account and returns status:

```
{"status":"OK"}
```

`POST v1/users/<username>/update` (author, admin)[1]

Example: `POST v1/users/alice/update`
Payload:

```
roles:  ["author","admin"] !
email:  "alice2@example.org"
new_username:  "alice2"
```

Updates user information and returns the new values:

```
{"status":"OK","user":{
  "username":"alice2",
  "name":"Alice Example",
  "email":"alice2@example.org",
  "image":"https://avatars.githubusercontent.com/u/1111111?v=3",
  "roles":["author","admin"],
  "identities":["github"]}
}
```

1.4.1 Current user

Part of Users resource is also used to gather information about currently signed-in user and manipulate him. However all initial information about user is given by the identity provider.

`GET v1/users/me` (user)

Returns information about currently signed-in user:

```
{"status":"OK","user":{
  "username":"bob",
  "name":"Bob Example",
  "email":"bob@example.org",
  "image":"https://lh3.googleusercontent.com/.../photo.jpg?sz=50",
  "roles":["author","admin"],
  "identities":["google"]}
}
```

`GET v1/users/me/signout` (user)

Signs out current user and clears current session. Returns status:

```
{"status":"OK"}
```

1.4.2 Roles

As described in section 1.2, there are currently two system-defined user roles – author and admin. However this can be changed in future. For information on roles in your instance of NARRA, please see:

`GET v1/users/roles` (author, admin)

```
{"status":"OK","roles":["admin","author"]}
```

[1] Only admin can change roles. Email and username can be changed by author as well.

1.5 System-related resources

To gather information about running instance of NARRA, `system`, `settings`, `generators`, `synthesizers` and `events` resources are available in the API.

1.5.1 System information

System resource provides basic information about NARRA instance.

`GET v1/system/version` (public)

Responds with version number of NARRA instance:

```
{"status":"OK","version":"dev"}
```

`GET v1/system/modules` (public)

Responds with list of modules (ruby gems) loaded added to NARRA core:

```
{"status":"OK",
"modules":[
  {"name":"narra-core",
   "version":"0.0.1",
   "summary":"NARRA Core functionality",
   "description":"NARRA Core functionality which covers all the NARRA data model,
    logic and SPI.",
   "authors":["Michal Mocnak","Krystof Pesek"],
   "email":["info@narra.eu"],
   "homepage":"http://www.narra.eu",
   "license":"GPL-3.0"},
  {"name":"narra-speech",
   "version":"0.0.1",
   "summary":"AT\u0026T Speech To Text",
   "description":"Speech To Text Generator using AT\u0026T Speech API to
    transcribe audio tracks",
   "authors":["Michal Mocnak"],
   "email":["info@narra.eu"],
   "homepage":"http://www.narra.eu",
   "license":"GPL-3.0"}
]}
```

1.5.2 Settings

Settings resource enables to view and modify system settings.

`GET v1/settings` (admin only)

Responds with list of current settings:

```
{"status":"OK","settings":[
  {"name":"storage_temp","value":"/tmp/narra"},
  {"name":"storage_local_path","value":"/opt/narra/storage"},
  {"name":"storage_local_endpoint","value":"http://narra-storage"},
  {"name":"storage_s3_items","value":"narra-items"},
  {"name":"thumbnail_extension","value":"png"},
```

```
  {"name":"thumbnail_resolution","value":"350x250"},
  {"name":"thumbnail_count","value":"5"},
  {"name":"video_proxy_extension","value":"webm"},
  {"name":"video_proxy_lq_resolution","value":"320x180"},
  {"name":"video_proxy_lq_bitrate","value":"300"},
  {"name":"video_proxy_hq_resolution","value":"1280x720"},
  {"name":"video_proxy_hq_bitrate","value":"1000"},
  {"name":"image_proxy_extension","value":"png"},
  {"name":"image_proxy_lq_resolution","value":"320x180"},
  {"name":"image_proxy_hq_resolution","value":"1280x720"},
  {"name":"audio_proxy_extension","value":"ogg"},
  {"name":"audio_proxy_bitrate","value":"112"},
  {"name":"items_probe_interval","value":"60"}
]}
```

`GET v1/settings/defaults` (user)

Responds with system default settings in different format:


```
{"status":"OK","defaults":{
  "storage_temp":"/tmp/narra",
  "storage_local_path":"/opt/narra/storage",
  "storage_local_endpoint":"http://narra-storage",
  "storage_s3_items":"narra-items",
  "thumbnail_extension":"png",
  "thumbnail_resolution":"350x250",
  "thumbnail_count":"5",
  "video_proxy_extension":"webm",
  "video_proxy_lq_resolution":"320x180",
  "video_proxy_lq_bitrate":"300",
  "video_proxy_hq_resolution":"1280x720",
  "video_proxy_hq_bitrate":"1000",
  "image_proxy_extension":"png",
  "image_proxy_lq_resolution":"320x180",
  "image_proxy_hq_resolution":"1280x720",
  "audio_proxy_extension":"ogg",
  "audio_proxy_bitrate":"112",
  "items_probe_interval":"60"}}
```

`GET v1/settings/<name>` (user)

Example: `GET v1/settings/storage_temp`

Responds with current value of one selected setting:


```
{"status":"OK","setting":{"name":"storage_temp","value":"/tmp/narra"}}
```

`POST v1/settings/<name>/update` (admin only)

Example: `POST v1/settings/storage_temp/update`
Payload:
`value: "/tmp/narra" !`

Updates a value of one selected setting and returns status:


```
{"status":"OK"}
```

1.5.3 Generators

Generators are modules or integral parts of NARRA, that provide easily processable data from multimedia content. For example audio transcription, recognition of any kind, classification of any kind etc.

`GET v1/generators` (author, admin)

Responds with a list of all generators currently running on NARRA:


```
{"status":"OK","generators":[
  {"identifier":"att_speech",
   "title":"AT\u0026T Speech To Text",
   "description":"Speech To Text Generator using AT\u0026T Speech API to
    transcribe audio tracks"
  }
]}
```


1.5.4 Synthesizers

Synthesizers are modules, that creates junctions between individual items in NARRA. Junction is a basic unit of Open Narrative principle, as it provides suggestions how the narration should or can continue.

`GET v1/synthesizers` (author, admin)

Responds with list of all synthesizers running on NARRA (currently none):


```
{"status":"OK","synthesizers":[]}
```


1.5.5 Events

NARRA uses an event-driven approach for item ingestion, metadata generation and basically all user actions that are assumed to take longer time to complete.

`GET v1/events` (admin only)

Responds with list of all currently running events:


```
{"status":"OK","events":[
  {"id":"5538da553533632e3c0d0000",
   "message":"narra::transcoder::5538da553533632e3c070000",
   "progress":0.3016603214890017,
   "status":"running",
   "item":{"id":"5538da553533632e3c070000","name":"0003H1-003","type":"video"}},
  {"id":"5548da553533632e3c0d0000",
   "message":"narra::transcoder::5548da553533632e3c070000",
   "progress":0.6937596037485183,
   "status":"running",
   "item":{"id":"5548da553533632e3c070000","name":"0004H1-003","type":"video"}}
]}
```

`GET v1/events/user/<username>` (author, admin)

Example: `GET v1/events/user/alice`

Responds with list of events associated with selected user:

```
{"status":"OK","events":[
  {"id":"5538da553533632e3c0d0000",
   "message":"narra::transcoder::5538da553533632e3c070000",
   "progress":0.3016603214890017,
   "status":"running",
   "item":{"id":"5538da553533632e3c070000","name":"0003H1-003","type":"video"}}
]}
```


1.6 Item-related resources

Item is the individual video clip, image, sound file or any piece of data supported by NARRA. Item is a building block of Open Narrative as well as finalized multimedia sequences.

Every item has to be organized in library.

GET v1/items (admin)

Responds with list of all items in NARRA instance:


```
{"status":"OK","items":[
  {"id":"552a33df61633249940c0000",
   "name":"00129O-051",
   "url":"http://example.org/00129O-051.mov",
   "type":"video",
   "prepared":true,
   "thumbnails":[
    "http://narra-storage/my-narra-items/552a33df61633249940c0000/thumbnail_00000.png"
   ],
   "video_proxy_hq":
    "http://narra-storage/my-narra-items/552a33df61633249940c0000/video_proxy_hq.webm",
   "video_proxy_lq":
    "http://narra-storage/my-narra-items/552a33df61633249940c0000/video_proxy_lq.webm",
   "audio_proxy":
    "http://narra-storage/my-narra-items/552a33df61633249940c0000/audio_proxy.ogg"},
  {"id":"552a33df61633276b1090000",
   "name":"0013D2-013-2",
   "url":"http://example.org/0013D2-013-2.mov",
   "type":"video",
   "prepared":true,
   "thumbnails":[
    "http://narra-storage/my-narra-items/552a33df61633276b1090000/thumbnail_00000.png",
    "http://narra-storage/my-narra-items/552a33df61633276b1090000/thumbnail_00004.png"
   ],
   "video_proxy_hq":
    "http://narra-storage/my-narra-items/552a33df61633276b1090000/video_proxy_hq.webm",
   "video_proxy_lq":
    "http://narra-storage/my-narra-items/552a33df61633276b1090000/video_proxy_lq.webm",
   "audio_proxy":
    "http://narra-storage/my-narra-items/552a33df61633276b1090000/audio_proxy.ogg"}
]}
```

GET v1/items/<id> (author, admin)

Example: `GET v1/items/552a33df61633249940c0000`

Responds with all information about one specific item, including metadata:


```
{"status":"OK","item":{
  "id":"552a33df61633249940c0000",
  "name":"001290-051",
  "url":"http://example.org/001290-051.mov",
  "type":"video",
  "prepared":true,
  "library":{"id":"552a328961633276b1000000","name":"Example Library"},
  "thumbnails":[
   "http://narra-storage/my-narra-items/552a33df61633249940c0000/thumbnail_00000.png"
  ],
  "video_proxy_hq":
   "http://narra-storage/my-narra-items/552a33df61633249940c0000/video_proxy_hq.webm",
  "video_proxy_lq":
   "http://narra-storage/my-narra-items/552a33df61633249940c0000/video_proxy_lq.webm",
  "audio_proxy":
   "http://narra-storage/my-narra-items/552a33df61633249940c0000/audio_proxy.ogg",
  "metadata":[
   {"name":"type","value":"video","generator":"source"},
   {"name":"name","value":"001290-051","generator":"source"},
   {"name":"url","value":"http://example.org/001290-051.mov","generator":"source"},
   {"name":"library","value":"Example Library","generator":"source"},
   {"name":"author","value":"Bob Example","generator":"source"},
   {"name":"thumbnail_00000",
    "value":"http://narra-storage/my-narra-items/552a33df61633249940c0000/thumbnail_00000.png",
    "generator":"thumbnail",
    "marks":[{"in":0.0}]},
   {"name":"video_proxy_lq",
    "value":"http://narra-storage/my-narra-items/552a33df61633249940c0000/video_proxy_lq.webm",
    "generator":"transcoder"},
   {"name":"video_proxy_hq",
    "value":"http://narra-storage/my-narra-items/552a33df61633249940c0000/video_proxy_hq.webm",
    "generator":"transcoder"},
   {"name":"size","value":"1305753","generator":"source"},
   {"name":"duration","value":"2.88","generator":"source"},
   {"name":"timecode","value":"12:19:20:00","generator":"source"},
   {"name":"bitrate","value":"3627","generator":"source"},
   {"name":"video_codec","value":"h264 (Main) (avc1 / 0x31637661)","generator":"source"},
   {"name":"colorspace","value":"yuv420p(tv, bt709)","generator":"source"},
   {"name":"resolution","value":"1280x720","generator":"source"},
   {"name":"width","value":"1280","generator":"source"},
   {"name":"height","value":"720","generator":"source"},
   {"name":"frame_rate","value":"25.0","generator":"source"},
   {"name":"audio_codec","value":"aac (LC) (mp4a / 0x6134706D)","generator":"source"},
   {"name":"audio_sample_rate","value":"48000","generator":"source"},
   {"name":"audio_channels","value":"2","generator":"source"},
   {"name":"audio_proxy",
    "value":"http://narra-storage/my-narra-items/552a33df61633249940c0000/audio_proxy.ogg",
    "generator":"transcoder"}
  ]
}}
```

POST v1/items/new (author, admin)

Example: POST v1/items/new
Payload:

```
url:  "http://example.org/003290-051.mov"!
library:  "552a328961633276b1000000" !
author:  "Camera Guy"
metadata:  {"description":  "Some interesting description"}
```

Creates new item in selected library, starts item ingestion (thumbnail and proxy files generation). Responds with basic information about currently created item:


```
{"status":"OK","item":{
  "id":"552a338961633277b1000000",
  "name":"003290-051",
  "url":"http://example.org/003290-051.mov",
  "type":"video",
  "prepared":false,
  "library":{"id":"552a328961633276b1000000","name":"Example Library"},
  "metadata":[
   {"name":"type","value":"video","generator":"source"},
   {"name":"name","value":"003290-051","generator":"source"},
   {"name":"url","value":"http://example.org/003290-051.mov","generator":"source"},
   {"name":"library","value":"test","generator":"source"},
   {"name":"author","value":"testovaci","generator":"source"},
   {"name":"description","value":"test","generator":"bob"}
  ]
}}
```

GET v1/items/<id>/delete (author, admin)

Example: GET v1/items/552a338961633277b1000000/delete

Deletes selected item.

1.6.1 Item events

Events, mentioned in 1.5.5 can be gathered also for individual items:

GET v1/items/<id>/events (author, admin)

Example: GET v1/items/5538da553533632e3c070000/events


```
{"status":"OK","events":[
  {"id":"5538da553533632e3c0d0000",
   "message":"narra::transcoder::5538da553533632e3c070000",
   "progress":0.3016603214890017,
   "status":"running",
   "item":{"id":"5538da553533632e3c070000","name":"0003H1-003","type":"video"}}
]}
```


1.6.2 Item generators

Generators, mentioned in 1.5.3, can be run again on individual items:

GET v1/items/<id>/regenerate/<generator> (author, admin)

Example: GET v1/items/5538da553533632e3c070000/regenerate/att_speech

Runs again a metadata generator and responds with status message:

{"status":"OK"}

1.6.3 Item metadata

Metadata describe the contents of item, technical parameters or provide basically any human-readable and computer-processable information about the item.

Metadata can be either generated on import of the item or added / edited afterwards.

GET v1/items/<id>/metadata (author, admin)

Example: GET v1/items/552a33df61633249940c0000/metadata

Returns all metadata attached to selected item:


```
{"status":"OK","metadata":[
   {"name":"type","value":"video","generator":"source"},
   {"name":"name","value":"00129D-051","generator":"source"},
   {"name":"url","value":"http://example.org/00129D-051.mov","generator":"source"},
   {"name":"library","value":"Example Library","generator":"source"},
   {"name":"author","value":"Bob Example","generator":"source"},
   {"name":"thumbnail_00000",
    "value":"http://narra-storage/my-narra-items/552a33df61633249940c0000/thumbnail_00000.png",
    "generator":"thumbnail",
    "marks":[{"in":0.0}]},
   {"name":"video_proxy_lq",
    "value":"http://narra-storage/my-narra-items/552a33df61633249940c0000/video_proxy_lq.webm",
    "generator":"transcoder"},
   {"name":"video_proxy_hq",
    "value":"http://narra-storage/my-narra-items/552a33df61633249940c0000/video_proxy_hq.webm",
    "generator":"transcoder"},
   {"name":"size","value":"1305753","generator":"source"},
   {"name":"duration","value":"2.88","generator":"source"},
   {"name":"timecode","value":"12:19:20:00","generator":"source"},
   {"name":"bitrate","value":"3627","generator":"source"},
   {"name":"video_codec","value":"h264 (Main) (avc1 / 0x31637661)","generator":"source"},
   {"name":"colorspace","value":"yuv420p(tv, bt709)","generator":"source"},
   {"name":"resolution","value":"1280x720","generator":"source"},
   {"name":"width","value":"1280","generator":"source"},
   {"name":"height","value":"720","generator":"source"},
   {"name":"frame_rate","value":"25.0","generator":"source"},
   {"name":"audio_codec","value":"aac (LC) (mp4a / 0x6134706D)","generator":"source"},
   {"name":"audio_sample_rate","value":"48000","generator":"source"},
   {"name":"audio_channels","value":"2","generator":"source"},
   {"name":"audio_proxy",
    "value":"http://narra-storage/my-narra-items/552a33df61633249940c0000/audio_proxy.ogg",
    "generator":"transcoder"}
]}
```

GET v1/items/<id>/metadata/<meta>?generator=<generator> (author, admin)

Example: GET v1/items/552a33df61633249940c0000/metadata/thumbnail_00000?generator=thumbnail

Returns value of one selected meta of one item.


```
{"status":"OK","metadata":{
  "name":"thumbnail_00000",
  "value":"http://narra-storage/my-narra-items/552a33df61633249940c0000/thumbnail_00000.png",
  "generator":"thumbnail",
  "marks":[{"in":0.0}]
}}
```

`POST v1/items/<id>/metadata/new` (author, admin)

Example: `POST v1/items/552a33df61633249940c0000/metadata/new`
Payload:

```
meta:  "thumbnail_00004" !
value:  "http://narra-storage/my-narra-items/tmp/thumbnail_00004.png" !
generator:  "thumbnail" !
marks:  [{"in":  4.0}]
```

Adds new metadata to the selected item and returns status and added value:


```
{"status":"OK","metadata":{
  "name":"thumbnail_00004",
  "value":"http://narra-storage/my-narra-items/tmp/thumbnail_00004.png",
  "generator":"thumbnail",
  "marks":[{"in":4.0}]
}}
```

`POST v1/items/<id>/metadata/<meta>/update` (author, admin)

Example: `POST v1/items/552a33df61633249940c0000/metadata/thumbnail_00004/update`
Payload:

```
value:  "http://narra-storage/my-narra-items/final/thumbnail_00004.png" !
generator:  "thumbnail" !
marks:  [{"in":  4.5}]
```

Updates selected metadata for chosen item and returns new value of this metadata:


```
{"status":"OK","metadata":{
  "name":"thumbnail_00004",
  "value":"http://narra-storage/my-narra-items/final/thumbnail_00004.png",
  "generator":"thumbnail",
  "marks":[{"in":4.5}]
}}
```

`GET v1/items/<id>/metadata/<meta>/delete?generator=<generator>` (author, admin)

Example: `GET v1/items/552a33df61633249940c0000/metadata/thumbnail_00004/delete?generator=thumbnail`

Deletes selected metadata for chosen item and returns status:


```
{"status":"OK"}
```


1.6.4 Item thumbnails

For easy retrieval of random thumbnails, a dedicated resource is available.

Please note that thumbnails are in principle metadata attached to the item with generator "thumbnail".

GET v1/items/thumbnails/<count> (public)

Example: GET v1/items/thumbnails/2


```
{"status":"OK","thumbnails":[
  "http://narra-storage/my-narra-items/5536e3a33533636175070000/thumbnail_00020.png",
  "http://narra-storage/my-narra-items/5536dfc23533635f4e010000/thumbnail_00004.png"
]}
```


1.7 Library-related resources

Library is a basic way how to organize individual items into easily manageable units. Library can be shared across multiple projects and multiple users, herein called contributors.

GET v1/libraries (author, admin)

Returns list of all accessible libraries:


```
{"status":"OK","libraries":[
  {"id":"5538da453533632e3c060000",
   "name":"Example Library",
   "description":"This is an example",
   "author":{"username":"alice","name":"Alice Example"},
   "thumbnails":[
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00012.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00006.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00018.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00030.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00024.png"
   ],
   "contributors":[{"username":"bob","name":"Bob Example"}]},
  {"id":"5838da453533632e3c060000",
   "name":"My example",
   "description":"This is my example",
   "author":{"username":"bob","name":"Bob Example"},
   "thumbnails":[
    "http://narra-storage/my-narra-items/583a00e33533631e90000000/thumbnail_00006.png",
    "http://narra-storage/my-narra-items/583a00e33533631e90000000/thumbnail_00012.png",
   ],
   "contributors":[]}
]}
```

GET v1/libraries/<id> (author, admin)

Example: GET v1/libraries/5538da453533632e3c060000

Returns all information about selected library:


```
{"status":"OK","library":{
  "id":"5538da453533632e3c060000",
  "name":"Example Library",
  "description":"This is an example",
  "generators":[
   {"identifier":"att_speech",
    "title":"AT&T Speech To Text",
    "description":"Speech To Text Generator using AT&T Speech API to transcribe audio tracks"}
```

```
  ],
  "author":{"username":"alice","name":"Alice Example"},
  "thumbnails":[
   "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00012.png",
   "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00006.png",
   "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00018.png",
   "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00030.png",
   "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00024.png"
  ],
  "contributors":[{"username":"bob","name":"Bob Example"}],
  "projects":[
   {"id":"552a31cd6163324994020000",
    "name":"my_project",
    "title":"Example Project",
    "author":{"username":"bob","name":"Bob Example"}}
   ],
  "metadata":[
   {"name":"Location","value":"FAMU"},
   {"name":"Source","value":"SD_0320"}
  ]
}}
```

`POST v1/libraries/new` (author, admin)

Example: `POST v1/libraries/new`
Payload:

```
name:  "Example 2" !
author:  "alice"
contributors:  ["bob"]
generators:  ["att_speech"]
description:  "Another example library"
project:  "my_project"
```

Creates a new library and returns all information:


```
{"status":"OK","library":{
  "id":"553e76853533637a41010000",
  "name":"Example 2",
  "description":"Another example library",
  "generators":[
   {"identifier":"att_speech",
    "title":"AT&T Speech To Text",
    "description":"Speech To Text Generator using AT&T Speech API to transcribe audio tracks"}
  ],
  "author":{"username":"alice","name":"Alice Example"},
  "contributors":[{"username":"bob","name":"Bob Example"}],
  "projects":[
   {"id":"552a31cd6163324994020000",
    "name":"my_project",
    "title":"Example Project",
    "author":{"username":"bob","name":"Bob Example"}}
   ],
  "metadata":[]
}}
```

`POST v1/libraries/<id>/update` (author, admin)

Example: `POST v1/libraries/553e76853533637a41010000/update`
Payload:

```
name:  "Castle"
author:  "bob"
contributors:  ["bob","alice"]
generators:  []
description:  "Castle footage"
project:  "my_project"
```

Updates information about the selected library and returns new information:


```
{"status":"OK","library":{
  "id":"553e76853533637a41010000",
  "name":"Castle",
  "description":"Castle footage",
  "generators":[],
  "author":{"username":"bob","name":"Bob Example"},
  "contributors":[
   {"username":"bob","name":"Bob Example"},
   {"username":"alice","name":"Alice Example"}
  ],
  "projects":[
   {"id":"552a31cd6163324994020000",
    "name":"my_project",
    "title":"Example Project",
    "author":{"username":"bob","name":"Bob Example"}}
   ],
  "metadata":[]
}}
```

`GET v1/libraries/<id>/delete` (author, admin)

Example: `GET v1/libraries/553e76853533637a41010000/delete`

Deletes the selected library with all its content. Returns only status:


```
 {"status":"OK"}
```


1.7.1 Library generators

As a generator can be assigned to our library and the generator itself can extend its features, it makes sense to sometimes regenerate the data generated by the selected generator.

`GET v1/libraries/<id>/regenerate/<generator>` (author, admin)

Example: `GET v1/libraries/5538da453533632e3c060000/regenerate/att_speech`

Deletes and starts generation of new metadata generated by one selected generator. Returns system status:


```
{"status":"OK"}
```

1.7.2 Library items

A list of items organized in selected library can be retrieved as well. Either as a whole list or a limited list.

`GET v1/libraries/<id>/items?limit=<limit>` (author, admin)

Example: `GET v1/libraries/5538da453533632e3c060000/items`

Returns list of items within selected library:


```
{"status":"OK","items":[
  {"id":"553a00e33533631e90000000",
   "name":"0003H1-003",
   "url":"http://example.org/0003H1-003.mov",
   "type":"video",
   "prepared":true,
   "thumbnails":[
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00018.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00030.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00024.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00012.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00006.png"
   ],
   "video_proxy_hq":
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/video_proxy_hq.webm",
   "video_proxy_lq":
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/video_proxy_lq.webm",
   "audio_proxy":
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/audio_proxy.ogg"}
]}
```


1.7.3 Library metadata

Metadata can be added to whole libraries, however they are not meant to be propagated to individual items, so they would not be possibly available in editing software connected to Narra.

`GET v1/libraries/<id>/metadata` (author, admin)

Example: `GET v1/libraries/5538da453533632e3c060000/metadata`

Returns all metadata attached to library:


```
{"status":"OK","metadata":[
  {"name":"Location","value":"FAMU"},
  {"name":"Source","value":"SD_0320"}
]}
```

`GET v1/libraries/<id>/metadata/<meta>` (author, admin)

Example: `GET v1/libraries/5538da453533632e3c060000/metadata/Location`

Returns value of one selected metadata:


```
{"status":"OK","metadata":{"name":"Location","value":"FAMU"}}
```

`POST v1/libraries/<id>/metadata/new` (author, admin)

Example: POST v1/libraries/5538da453533632e3c060000/metadata/new
Payload:
meta: "Camera" !
value: "Dave Example" !

Adds new metadata and returns only the sent one:

{"status":"OK","metadata":{"name":"Camera","value":"Dave Example"}}

POST v1/libraries/<id>/metadata/<meta>/update (author, admin)

Example: POST v1/libraries/5538da453533632e3c060000/metadata/Camera/update
Payload:
value: "Zoe Example" !

Updates value of selected metadata and returns only the updated one:

{"status":"OK","metadata":{"name":"Camera","value":"Zoe Example"}}

GET v1/libraries/<id>/metadata/<meta>/delete (author, admin)

Example: GET v1/libraries/5538da453533632e3c060000/metadata/Camera/delete

Deletes the selected metadata and returns status:

{"status":"OK"}

1.8 Project-related resources

Project is a top level of item organization. The main purpose of project is to keep organized all media involved in single piece of audiovisual work.

Whole project can be shared across contributors or even made fully public.

GET v1/projects (public)

Returns list of all projects:


```
{"status":"OK","projects":[
  {"name":"my_project",
   "title":"Example Project",
   "description":"This is an example project.",
   "author":{"username":"bob","name":"Bob Example"},
   "public":"false",
   "thumbnails":[
    "http://narra-storage/my-narra-items/5536dfc23533635f4e010000/thumbnail_00012.png",
    "http://narra-storage/my-narra-items/552a33df6163324994130000/thumbnail_00020.png",
    "http://narra-storage/my-narra-items/552a33df6163324994050000/thumbnail_00018.png",
    "http://narra-storage/my-narra-items/5536e3a335336361ae000000/thumbnail_00008.png",
    "http://narra-storage/my-narra-items/552a33df6163324994210000/thumbnail_00020.png"
   ],
   "contributors":[
    {"username":"bob","name":"Bob Example"},
    {"username":"alice","name":"Alice Example"}
   ]},
  {"name":"another",
   "title":"Another Example",
   "description":"This is yet another example project.",
```

```
   "author":{"username":"bob","name":"Bob Example"},
   "public":"false",
   "thumbnails":[
    "http://narra-storage/my-narra-items/5536dfc23533635f4e010000/thumbnail_00012.png",
    "http://narra-storage/my-narra-items/552a33df6163324994130000/thumbnail_00020.png",
    "http://narra-storage/my-narra-items/552a33df6163324994050000/thumbnail_00018.png",
    "http://narra-storage/my-narra-items/5536e3a335336361ae000000/thumbnail_00008.png",
    "http://narra-storage/my-narra-items/552a33df6163324994210000/thumbnail_00020.png"
   ],
   "contributors":[]}
]}
```

GET v1/projects/<name> (admin, author)

Example: GET v1/projects/my_project

Returns all properties of selected project:


```
{"status":"OK","project":{
  "name":"my_project",
  "title":"Example Project",
  "description":"This is an example project.",
  "author":{"username":"bob","name":"Bob Example"},
  "public":"false",
  "thumbnails":[
   "http://narra-storage/my-narra-items/5536dfc23533635f4e010000/thumbnail_00012.png",
   "http://narra-storage/my-narra-items/552a33df6163324994130000/thumbnail_00020.png",
   "http://narra-storage/my-narra-items/552a33df6163324994050000/thumbnail_00018.png",
   "http://narra-storage/my-narra-items/5536e3a335336361ae000000/thumbnail_00008.png",
   "http://narra-storage/my-narra-items/552a33df6163324994210000/thumbnail_00020.png"
  ],
  "contributors":[
   {"username":"bob","name":"Bob Example"},
   {"username":"alice","name":"Alice Example"}
  ],
  "libraries":[
   {"id":"5538da453533632e3c060000",
    "name":"Example Library",
    "description":"This in an example",
    "author":{"username":"alice","name":"Alice Example"},
    "thumbnails":[
     "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00012.png",
     "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00006.png",
     "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00018.png",
     "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00030.png",
     "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00024.png"
    ],
    "contributors":[{"username":"bob","name":"Bob Example"}]}
  ],
  "metadata":[
   {"name":"public","value":"false"},
   {"name":"Editor","value":"Zoe Example"}
  ]
}}
```

`POST v1/projects/new` (author, admin)

Example: `POST v1/projects/new`
Payload:

```
name:  "old_prague" !
title:  "Old Prague Documentary" !
author:  "bob"
contributors:  ["alice"]
description:  "Some documentary"
```

Creates a new project and returns information about it:

```
{"status":"OK","project":{
  "name":"old_prague",
  "title":"Old Prague Documentary",
  "description":"Some wonderful documentary",
  "author":{"username":"bob","name":"Bob Example"},
  "public":"false",
  "contributors":[{"username":"alice","name":"Alice Example"}],
  "libraries":[],
  "metadata":[{"name":"public","value":"false"}]
}}
```

`POST v1/projects/<name>/update` (author, admin)

Example: `POST v1/projects/old_prague/update`
Payload:

```
new_name:  "prague"
title:  "Prague Documentary"
author:  "alice"
contributors:  ["alice","bob"]
description:  "Some documentary"
```

Updates attributes of the project and returns information about updated project:

```
{"status":"OK","project":{
  "name":"prague",
  "title":"Prague Documentary",
  "description":"Some documentary",
  "author":{"username":"alice","name":"Alice Example"},
  "public":"false",
  "contributors":[
   {"username":"alice","name":"Alice Example"}
   {"username":"bob","name":"Bob Example"}
  ],
  "libraries":[],
  "metadata":[{"name":"public","value":"false"}]
}}
```

`GET v1/projects/<name>/delete` (author, admin)

Example: `GET v1/projects/prague/delete`

Deletes the project (libraries are kept but can be orphaned and become inaccessible) and returns status:

```
{"status":"OK"}
```

1.8.1 Project items

Although items are organized in libraries and that is a preferred way how to retrieve them, project can list them as well.

`GET v1/projects/<name>/items` (author, admin)

Example: `GET v1/project/my_project/items`

Returns list of items in project, note that the structure is the same as a list of library items:


```
{"status":"OK","items":[
  {"id":"553a00e33533631e90000000",
   "name":"0003H1-003",
   "url":"http://example.org/0003H1-003.mov",
   "type":"video",
   "prepared":true,
   "thumbnails":[
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00018.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00030.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00024.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00012.png",
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00006.png"
   ],
   "video_proxy_hq":
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/video_proxy_hq.webm",
   "video_proxy_lq":
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/video_proxy_lq.webm",
   "audio_proxy":
    "http://narra-storage/my-narra-items/553a00e33533631e90000000/audio_proxy.ogg"}
]}
```


1.8.2 Project libraries

Because the connection between projects and libraries is many-to-many, it makes sense to manage adding or removing libraries from the project side as well.

`POST v1/projects/<name>/libraries/<action>` (author, admin)

Example: `POST v1/projects/my_project/libraries/remove`
Payload:
`libraries: ["Example Library"] !`

Removes libraries form project (libraries themselves are not deleted) and returns information about whole project:


```
{"status":"OK","project":{
  "name":"my_project",
  "title":"Example Project",
  "description":"This is an example project.",
  "author":{"username":"bob","name":"Bob Example"},
  "public":"false",
  "thumbnails":[],
  "contributors":[
   {"username":"bob","name":"Bob Example"},
   {"username":"alice","name":"Alice Example"}
  ],
```

```
  "libraries":[],
  "metadata":[
   {"name":"public","value":"false"},
   {"name":"Editor","value":"Zoe Example"}
  ]
}}
```


Example: POST v1/projects/my_project/libraries/add
Payload:
libraries: ["Example Library"] !

Adds libraries to project and returns information about whole project:


```
{"status":"OK","project":{
  "name":"my_project",
  "title":"Example Project",
  "description":"This is an example project.",
  "author":{"username":"bob","name":"Bob Example"},
  "public":"false",
  "thumbnails":[
   "http://narra-storage/my-narra-items/5536dfc23533635f4e010000/thumbnail_00012.png",
   "http://narra-storage/my-narra-items/552a33df6163324994130000/thumbnail_00020.png",
   "http://narra-storage/my-narra-items/552a33df6163324994050000/thumbnail_00018.png",
   "http://narra-storage/my-narra-items/5536e3a335336361ae000000/thumbnail_00008.png",
   "http://narra-storage/my-narra-items/552a33df6163324994210000/thumbnail_00020.png"
  ],
  "contributors":[
   {"username":"bob","name":"Bob Example"},
   {"username":"alice","name":"Alice Example"}
  ],
  "libraries":[
   {"id":"5538da453533632e3c060000",
    "name":"Example Library",
    "description":"This in an example",
    "author":{"username":"alice","name":"Alice Example"},
    "thumbnails":[
     "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00012.png",
     "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00006.png",
     "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00018.png",
     "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00030.png",
     "http://narra-storage/my-narra-items/553a00e33533631e90000000/thumbnail_00024.png"
    ],
    "contributors":[{"username":"bob","name":"Bob Example"}]}
  ],
  "metadata":[
   {"name":"public","value":"false"},
   {"name":"Editor","value":"Zoe Example"}
  ]
}}
```


1.8.3 Project metadata

Metadata attached to project are again not propagated to libraries, neither items. Therefore these metadata should only describe things connected to the project itself.

GET v1/projects/<name>/metadata (author, admin)

Example: GET v1/projects/my_project/metadata

Returns list of all metadata:

```
{"status":"OK","metadata":[
  {"name":"public","value":"false"},
  {"name":"Editor","value":"Zoe Example"}
]}
```

GET v1/projects/<name>/metadata/<meta> (author, admin)

Example: GET v1/projects/my_project/metadata/Editor

Returns value of selected metadata:

```
{"status":"OK","metadata":{"name":"Editor","value":"Zoe Example"}}
```

POST v1/projects/<name>/metadata/new (author, admin)

Example: POST v1/projects/my_project/metadata/new
Payload:

```
meta:  "Started" !
value:  "21.  2.  2014" !
```

Creates new project metadata and returns added metadata:

```
{"status":"OK","metadata":{"name":"Started","value":"21. 2. 2014"}}
```

POST v1/projects/<name>/metadata/<meta>/update (author, admin)

Example: POST v1/projects/my_project/metadata/Started/update
Payload:

```
value:  "20.  2.  2014" !
```

Updates selected metadata and returns it:

```
{"status":"OK","metadata":{"name":"Started","value":"20. 2. 2014"}}
```

GET v1/projects/<name>/metadata/<name>/delete (author, admin)

Example: GET v1/projects/my_project/metadata/Started/delete

Deletes selected metadata from project and returns status:

```
{"status":"OK"}
```

1.8.4 Project sequences

Standard video editing process involves a sequences (sometimes called timelines) that contain information about what parts of multimedia footage are used and in what order. Although this is not standard to Open Narrative structures, most editors are familiar with this approach and require it as a possible workflow for adding new item connections.

This feature is currently under development and will be changed a lot in future releases. Currently there is also no way how to add sequences to NARRA and thus the list is empty:

GET v1/projects/<name>/sequences (admin, author)

Example: GET v1/projects/my_project/sequences

{"status":"OK","sequences":[]}