ListenBrainz Documentation

Release 0.1.0

MetaBrainz Foundation

Jul 28, 2017

Contents

This documentation is aimed at developers who want to *use our API* to submit and fetch listens. We suggest taking a look the the *json* documentation, since most of the complexity in using the API comes from constructing/reading the ListenBrainz JSON documents.

Contents:

CHAPTER 1

ListenBrainz API for Beta

The ListenBrainz server supports the following end-points for submitting and fetching listens. All endpoints have this root URL for our current "pre-beta" site:

Beta API Root URL: `https://beta-api.listenbrainz.org` **Beta Web Root URL**: `https://beta.listenbrainz.org`

Once we go to a full beta, we're going to move to the production URLs:

Production API Root URL: `https://api.listenbrainz.org` **Production Web Root URL**: `https://listenbrainz.org`

The current site at listenbrainz.org will move to alpha.listenbrainz.org once we're in beta.

NOTE: All of ListenBrainz services are available on **HTTPS** only!

Reference

API Calls

`POST /1/submit-listens`

Submit listens to the server. A user token (found on https://listenbrainz.org/user/import) must be provided in the Authorization header!

For complete details on the format of the JSON to be POSTed to this endpoint, see *JSON Documentation*.

Request Headers

- Authorization – Token <user token>

Status Codes

- 200 OK – listen(s) accepted.
- 400 Bad Request – invalid JSON sent, see error message for details.
- 401 Unauthorized – invalid authorization. See error message for details.

Response Headers

- Content-Type – *application/json*

`GET /1/latest-import`

Get and update the timestamp of the newest listen submitted in previous imports to ListenBrainz.

In order to get the timestamp for a user, make a GET request to this endpoint. The data returned will be JSON of the following format:

{ 'musicbrainz_id': the MusicBrainz ID of the user,

'latest_import': the timestamp of the newest listen submitted in previous imports. Defaults to 0

}

Parameters

- **user_name** – the MusicBrainz ID of the user whose data is needed

Status Codes

- 200 OK – Yay, you have data!

Response Headers

- Content-Type – *application/json*

In order to update the timestamp of a user, you'll have to provide a user token in the Authorization Header. User tokens can be found on https://listenbrainz.org/user/import .

The JSON that needs to be posted must contain a field named *ts* in the root with a valid unix timestamp.

Request Headers

- Authorization – Token <user token>

Status Codes

- 200 OK – latest import timestamp updated
- 400 Bad Request – invalid JSON sent, see error message for details.
- 401 Unauthorized – invalid authorization. See error message for details.

POST /1/latest-import

Get and update the timestamp of the newest listen submitted in previous imports to ListenBrainz.

In order to get the timestamp for a user, make a GET request to this endpoint. The data returned will be JSON of the following format:

{ 'musicbrainz_id': the MusicBrainz ID of the user,

'latest_import': the timestamp of the newest listen submitted in previous imports. Defaults to 0

}

Parameters

- **user_name** – the MusicBrainz ID of the user whose data is needed

Status Codes

- 200 OK – Yay, you have data!

Response Headers

- Content-Type – *application/json*

In order to update the timestamp of a user, you'll have to provide a user token in the Authorization Header. User tokens can be found on https://listenbrainz.org/user/import .

The JSON that needs to be posted must contain a field named *ts* in the root with a valid unix timestamp.

Request Headers

- Authorization – Token <user token>

Status Codes

- 200 OK – latest import timestamp updated
- 400 Bad Request – invalid JSON sent, see error message for details.
- 401 Unauthorized – invalid authorization. See error message for details.

GET /1/user/(*user_name*)**/listens**

Get listens for user `user_name`. The format for the JSON returned is defined in our *JSON Documentation*.

If none of the optional arguments are given, this endpoint will return the `DEFAULT_ITEMS_PER_GET` most recent listens. The optional `max_ts` and `min_ts` UNIX epoch timestamps control at which point in time to start returning listens. You may specify max_ts or min_ts, but not both in one call. Listens are always returned in descending timestamp order.

Parameters

- **max_ts** – If you specify a `max_ts` timestamp, listens with listened_at less than (but not including) this value will be returned.
- **min_ts** – If you specify a `min_ts` timestamp, listens with listened_at greater than (but not including) this value will be returned.
- **count** – Optional, number of listens to return. Default: `DEFAULT_ITEMS_PER_GET` . Max: `MAX_ITEMS_PER_GET`

Status Codes

- 200 OK – Yay, you have data!

Response Headers

- Content-Type – *application/json*

Timestamps

All timestamps used in this project are UNIX epoch timestamps in UTC. When submitting timestamps to us, please ensure that you have no timezone adjustments on your timestamps.

Constants

Constants that are releavant to using the API:

`listenbrainz.webserver.views.api_tools.`**MAX_LISTEN_SIZE = 10240**
Maximum overall listen size in bytes, to prevent egregious spamming.

`listenbrainz.webserver.views.api_tools.`**MAX_ITEMS_PER_GET = 100**
The maximum number of listens returned in a single GET request.

`listenbrainz.webserver.views.api_tools.`**DEFAULT_ITEMS_PER_GET = 25**
The default number of listens returned in a single GET request.

`listenbrainz.webserver.views.api_tools.`**MAX_TAGS_PER_LISTEN = 50**
The maximum number of tags per listen.

`listenbrainz.webserver.views.api_tools.`**MAX_TAG_SIZE = 64**
The maximum length of a tag

CHAPTER 2

JSON Documentation

Submission JSON

To submit a listen via our *ListenBrainz API for Beta* you'll need to POST a JSON document to the `submit-listens` endpoint. You can submit one of three types JSON documents:

- `single` - submit a single listen. This indicates that the user just finished listening to this track. Only a single track may be specified in the `payload`.
- `playing_now` - submit a playing_now notification; This indicates that the user just began listening to this track. The `payload` may contain only one track and submitting `playing_now` documents is optional.
- `import` - submit previously saved listens; the `payload` may contain more than one listen, but the complete document may not exceed `MAX_LISTEN_SIZE` bytes in size.

These different types of submissions are defined by the `listen_type` element at the top-most level of the submitted JSON document. The only other element required at the top level is the payload element that provides an array of listens – the payload may be one or more listens (as designated by the listen_type):

```
{
  "listen_type": "single",
  "payload": [
      --- listen data here ---
  ]
}
```

A sample listen payload may look like:

```
{
  "listened_at": 1443521965,
  "track_metadata": {
    "additional_info": {
      "release_mbid": "bf9e91ea-8029-4a04-a26a-224e00a83266",
      "artist_mbids": [
        "db92a151-1ac2-438b-bc43-b82e149ddd50"
      ],
      "recording_mbid": "98255a8c-017a-4bc7-8dd6-1fa36124572b",
      "tags": [ "you", "just", "got", "rick rolled!"]
    },
    "artist_name": "Rick Astley",
    "track_name": "Never Gonna Give You Up",
    "release_name": "Whenever you need somebody"
```

```
  }
}
```

A complete submit listen JSON document may look like:

```
{
  "listen_type": "single",
  "payload": [
    {
      "listened_at": 1443521965,
      "track_metadata": {
        "additional_info": {
          "release_mbid": "bf9e91ea-8029-4a04-a26a-224e00a83266",
          "artist_mbids": [
            "db92a151-1ac2-438b-bc43-b82e149ddd50"
          ],
          "recording_mbid": "98255a8c-017a-4bc7-8dd6-1fa36124572b",
          "tags": [ "you", "just", "got", "rick rolled!"]
        },
        "artist_name": "Rick Astley",
        "track_name": "Never Gonna Give You Up",
        "release_name": "Whenever you need somebody"
      }
    }
  ]
}
```

Fetching listen JSON

The JSON documents returned from our API look like the following:

```
{
  "count": 25,
  "payload": [
      "-- listen data here ---"
  ]
}
```

The top level count element indicates how many listens are returned in this document. The other element is the payload element, which contains listen JSON elements as described above.

Payload JSON details

A minimal payload must include the `listened_at`, `track_metadata/artist_name` and `track_metadata/track_name` elements:

```
{
  "listened_at": 1443521965,
  "track_metadata": {
    "artist_name": "Rick Astley",
    "track_name": "Never Gonna Give You Up",
  }
}
```

We strongly recommend to add whatever additional metadata you may have for a track to the `additional_info` element. Any additional information will allow us to better correlate your listen data to existing MusicBrainz based data. If you have MusicBrainz IDs availble, submit them!

The following optional elements may also be included in the `track_metadata` element:

element	description
`release_name`	the name of the release this recording was played from.

The following optional elements may also be included in the `additional_info` element. If you do not have the data for any of the following fields, omit the key entirely:

element	description
`artist_mbids`	A list of MusicBrainz Artist IDs, one or more Artist IDs may be included here. If you have a complete MusicBrainz artist credit that contains multiple Artist IDs, include them all in this list.
`release_group_mbid`	A MusicBrainz Release Group ID of the release group this recording was played from.
`release_mbid`	A MusicBrainz Release ID of the release this recording was played from.
`recording_mbid`	A MusicBrainz Recording ID of the recording that was played.
`track_mbid`	A MusicBrainz Track ID associated with the recording that was played.
`work_mbids`	A list of MusicBrainz Work IDs that may be associated with this recording.
`tracknumber`	The tracknumber of the recording. This first recording on a release is tracknumber 1.
`isrc`	The ISRC code associated with the recording.
`spotify_id`	The Spotify track URL associated with this recording. e.g.: http://open.spotify.com/track/1rrgWMXGCGHru5bIRxGFV0
`tags`	A list of user defined tags to be associated with this recording. These tags are similar to last.fm tags. For example, you have apply tags such as `punk`, `see-live`, `smelly`. You may submit up to `MAX_TAGS_PER_LISTEN` tags and each tag may be up to `MAX_TAG_SIZE` characters large.

At this point, we are not scrubbing any superflous elements that may be submitted via the `additional_info` element. We're open to see how people will make use of these unspecified fields and may decide to formally specify or scrub elements in the future.

Please do not submit copyrighted information in these fields!!

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

/1

```
GET /1/latest-import, 3
GET /1/user/(user_name)/listens, 5
POST /1/latest-import, 4
POST /1/submit-listens, 3
```

D

M