Описание АПИ

Общие положения

Базовый урл для выполнения запросов: **https://v2api.musicalligator.com/api**

АПИ построено на принципах REST и использует следующие HTTP методы: **GET, POST, PUT, PATCH, DELETE**.

АПИ принимает и отдает значения в формате **JSON** (за исключением методов загрузки и скачивания файлов). Возвращаемые значения приходят в поле **data** ответа.

При успешном запросе возвращаются статус коды **200** и **201**. При ошибках с данными возвращается ошибка **400** с кодом ошибки. При ошибке авторизации возвращается статус 401. При ошибке сервера возвращается статус **500**. При недоступности сервера возвращается статус **503**.

Авторизация и аутентификация

Авторизация осуществляется с помощью пары логин\пароль:

Метод: *POST*
URL: */auth/sign_in*
Data: { login: string; password: string; }
Response:

```
{
 sessionToken: string;
 account:
 {
 clientId: number;
 dtCreate: string | Date;
 username?: string;
 firstName?: string;
 lastName?: string;
 middleName?: string;
 role: ClientAccountRole;
 avatarUrl?: string;
 contractData: object;
 contractType: ContractType;
 status: ClientStatus;
 phone: string;
 session?: string;
 royalty: number;
 }
}
```

Для аутентификации необходимо использовать значение ***sessionToken***, а так же ***account.clientId*** для доступа к некоторым ресурсам.

Аутентификация

Аутентификация осуществляется путем передачи значения ***sessionToken*** в хедере **Authorization** и является обязательной для всех методов апи. Так же обязательным является хедер **X-Lang,** который может принимать значения **EN** или **RU,** передача этого хедера необходима для загрузки релизов в правильный кабинет и выставление правильной локализации.

Пример запроса с необходимыми хедерами:

curl -XGET -H 'Authorization: test-token' -H 'X-Lang: RU' -H "Content-type: application/json" 'https://v2api.musicalligator.com/api/releases/1'

Платформы

Платформы используются для получения всех доступных платформ, при этом список доступных платформ для отгрузки может включать не все активные платформы.

Метод: *GET*
URL: */platform/platforms/streaming*
Querystring: *{withProvidersOnly: boolean;}*
Response:
```
{
        id: number;
        name: string;
        publicName: string;
        urlIcon: string;
        releaseTypes: {releaseTypeId: number; releaseType: string;}
        domains: string[];
}
```

При передаче параметра withProvidersOnly=true, вернется список активных платформ, на которые можно отгружать релизы, которые необходимо указать при создании релиза. Так же не все платформы доступны на всех типах релизов, необходимо использовать массив ***releaseTypes*** для определения платформ, подходящих под нужный тип релиза.

Персоны

Персоны – люди, которые принимали участие в создание релиза. На данный момент поддерживается 3 типа персон: автор слов, автор музыыки, режиссер (для клипа).

Список персон

Метод: *GET*
URL: */persons*
Querystring: *{name?: string; ids?: number[]}*
Response:
```
{
        id: number;
        name: string;
        platforms:{id: number; name: string; platformId?: string;}[]
```

}

Метод возвращает список добавленных персон и их привязанные платформы. Можно сделать поиск по частичному совпадению имени персоны.

Создание персоны

При создании персоны необхдимо указывать полное ФИО.

Метод: *POST*
URL: */persons*
Data:
{
name: string;
platforms:{id: number; name: string; platformId?: string;}[]
}
Response: *{id: number}*

При передаче пустого значения paltformId, для персоны будет создана новая карточка на платформе. При указании карточек персоны на платформах доступны только платформы Spotify (id 291) и Apple Music (id 282).

Нельзя создавать персон с одинаковыми именами.

Обновление персоны

Метод: *PUT*
URL: */persons/PERSON_ID*
DATA:
{
name: string;
platforms:{id: number; name: string; platformId?: string;}[]
}

Обновляет данные персоны. При этом обновления по активным релизам, в которых участвует персона не отсылаются.

Артисты

Артисты – непосредственные исполнители треков в релизах. На данный момент в релизах и треках поддерживается 3 типа артистов: основной исполнителье, дополнительные исполнитель, фит.

Список артистов

Метод: *GET*
URL: /*artists*
Querystring: *{name?: string; ids?: number[]}*
Response:

{
id: number;
name: string;
platforms:{id: number; name: string; platformId?: string;}[]
}

Метод возвращает список добавленных артистов и их привязанные платформы. Можно сделать поиск по частичному совпадению имени артиста.

Создание артиста

Метод: *POST*
URL: */artists*
Data:
{
name: string;
platforms:{id: number; name: string; platformId?: string;}[]
}
Response: *{id: number}*

При передаче пустого значения paltformId, для артиста будет создана новая карточка на платформе. При указании карточек персоны на платформах доступны только платформы Spotify (id 291) и Apple Music (id 282).

Нельзя создавать артистов с одинаковыми именами.

Обновление артиста

Метод: *PUT*
URL: */artists/ARTIST_ID*
Data:
{
name: string;
platforms:{id: number; name: string; platformId?: string;}[]
}

Лейблы

Лейблы – фактические лейблы, под которыми выпускается релиз. У одного клиента может быть несколько лейблов.

Список лейблов

Метод: *GET*
URL: */labels*
Querystring: *{name?: string;}*
Response:
{

count: number;
data: {
id: number;
name: string;
level:string;
}[]
}

Метод возвращает список добавленных лейблов и их уровней. Можно сделать поиск по частичному совпадению имени лейбла.

Создание лейбла

Метод: *POST*
URL: */labels*
Data:
{
name: string;
}
Response: *{id: number; name: string}*

Создает новый лейбл для клиента. Если у пользователя такой лейбл уже существует, но написан в другом регистре, то вернется ошибка RELEASE_LABEL_BUSY. Так же проводится проверка на название самого лейбла, при некорректном названии вернется ошибка RELEASE_BAD_LABEL.

Обновление лейбла

Метод: *PUT*
URL: */labels/LABEL_ID*
Data:
{
name: string;
}

Обновляет название лейбла. При этом, если у лейбла есть выпущенные релизы или релизы ожидающие отгрузку, то обновить название лейбла нельзя и вернется ошибка RELEASE_IS_RELEASED.

Проверка названия лейбла

Метод: *POST*
URL: */labels/check*
Data:
{
name: string;
}

Проверяет название лейбла на соответствие требованиям и занятость. В случае ошибки вернет ошибки RELEASE_BAD_LABEL, RELEASE_LABEL_BUSY. В случае успеха вернется статус 201.

Релизы

Создание и обновление релиза

Создание релиза происходит в 2 этапа: сначала создается оболочка релиза, в которую потом добавляются необходимые данные через обновление релиза. Так же происходит создание и обновление трека. Для создания релиза так же необходимы id артистов и персон, а так же id жанров, id типа релиза и id языка треков.

Получение жанров, типов релизов, языков

Метод: *GET*
URL: */releases/filters*
Response:
{
types: {releaseTypeId: number; releaseType: string}[];
langs: {langId: number; langName: string; langCode: string;}[];
genres: {genreId: number; parentId: number | null; genreName: string; category: string}[]
}

Жанры возвращаются массивом, с указанием parentId, если он есть, то этот жанр является под жанром основного жанра и при создании релиза нужно указать жанр родитель как основной жанр.

Создание релиза

Метод: *POST*
URL: */releases/create*
Data: *{releaseType: string}*
Response: *{release: {releaseId: number; tracks: [{trackId: number}]}}*

При создании релиза в нем создается так же один трек, который можно наполнять данными. При создании релиза указывается строковое значение типа релиза, а не id типа.

Добавление нового трека к релизу

Метод: *POST*
URL: */releases/RELEASE_ID/tracks*
Response: *{trackId: number}*

К существующему релизу можно добавить треки. На кол-во треков в релизе есть ограничения, в зависимости от типа релиза:

Рингтон - 1 трек

Звук Тик-Ток - 1 трек

Клип - 1 трек

Сингл - 3 трека

Так же нельзя добавлять треки к выпущенному релизу и к релизу, который находится в статусе **DRAFT.**

Обновление данных релиза

Метод: *PUT*
URL: */releases/RELEASE_ID*
Data:
```
{
  title?: string;
  clineYear?: number;
  clineValue?: string;
  plineYear?: number;
  plineValue?: string;
  labelId?: number;
  ownEan?: string;
  releaseType?: string;
  artists?: {id: number; role: 'MAIN' | 'ADVANCED' | 'FEAT'}[];
  releaseDate?: string;
  originalReleaseDate?: string;
  genre?: {genreId: number; subGenreId?: number};
  additionalGenres?: {genreId: number; subGenreId?: number}[];
  language?: number;
  releaseVersion?: string;
  streamingPlatforms?: number[];
}
```
Response: *ReleaseObject* (см. получение релиза по ID)

Данные релиза можно обновлять как полностью, так и частично. Если слать значения, которые уже имеются в релизе, то они обновлены не будут. EAN релиза можно не передавать, если его нет, платформа сама присвоит EAN при выгрузке релиза на платформы. Можно указать основной жанр и 2 дополнительных жанра. У релиза может быть только 1 основной артист.

Загрузка обложки релиза

Метод: *POST*
URL: */releases/RELEASE_ID/cover*
Content-type: *multipart/form-data*
Parameters: *file to upload*

Обновляет обложку релиза. Поддерживаемые форматы файлов: image/jpg, image/jpeg, image/png.

Обложка для музыкального релиза должна быть не менее 3000x3000. Для клипа не менее 2560x1440 и соотношение сторон 16:9.

Обновление данных трека

Метод: *PUT*
URL: */releases/RELEASED_ID/tracks/TRACK_ID*
Data:

```
{
  title?: string;
  language?: number;
  genre?: {genreId: number; subGenreId?: number};
  additionalGenres?: {genreId: number; subGenreId?: number}[];
  startPoint?: number;
  artists?: {id: number; role: 'MAIN' | 'ADVANCED' | 'FEAT'}[];
  persons?: {id: number; role: 'MUSIC_AUTHOR' | 'LYRICS_AUTHOR' | 'DIRECTOR'}[];
  instrumental?: boolean;
  adult?: boolean;
  text?: string;
  ownIsrc?: string;
  trackVersion?: string;
  copyrights?: number;
  related?: number;
  karaoke?: {rows: {begin: number; end: number; text: string; _blank: boolean;}[]};
  recordingYear?: number;
  clipPreviewFrame?: number;
}
```

Данные трека можно обновлять как полностью, так и частично. Если слать значения, которые уже имеются в релизе, то они обновлены не будут.
Можно указать свой ISRC в поле **ownISRC**, если его нет, платформа сама назначит ISRC при отгрузке релиза.
Текст песни (поле **text**) и караоке (поле **karaoke**) не являются обязательными для отгрузки релиза.
Поле **clipPreviewFrame** используется только для типа релиза **CLIP**.
Начало прослушивания (поле **startPoint**) не может превышать длительность трека и должно быть меньше длительности на 30 секунд. Если длительность трека меньше 30 секунд, то нужно ставить значение 0.
Для инструментального трека (поле *instrumental* = *true*), слать персону с ролью **LYRICS_AUTHOR** не нужно, так как у инструментального трека не может быть автора слов.
У трека может быть только 1 основной артист.

Загрузка трека

Метод: *POST*
URL: */releases/RELEASE_ID/tracks/TRACK_ID/upload*
Content-type: *multipart/form-data*
Parameters: *file to upload*

Обновляет медиа файл трека. Поддерживаемые форматы: wav, mp4.
Формат mp4 поддерживается только для типа релиза CLIP.
Аудио файл должен иметь 2 канала, минимальный битрейт 16, семпл рейт 44100.
При окончании обработки файла трека, в данных трека поле **proc_status** примет значение **completed**. При ошибке, поле **proc_status** примет значение **error**, а в поле **proc_message** будет содержаться код ошибки.
Релиз нельзя отправить на отгрузку, пока все треки не обработаны.

Изменение порядка треков

Метод: *POST*
URL: */releases/RELEASED_ID/tracks/order*
Data: *{order: {trackId: number, trackOrder: number}[]}*

По умолчанию порядок треков соответствует порядку их создания через АПИ. Если нужно поменять порядок треков, то следует использовать этот метод. Параметр **trackOrder** должен содержать порядковый номер трека, без пропусков.

Удаление трека

Метод: *DELETE*
URL: */releases/RELEASE_ID/tracks/TRACK_ID*

Удаляет трек. Удалить трек в выпущенном релизе нельзя.

Удаление релиза

Метод: *DELETE*
URL: */releases/RELEASE_ID*

Удаляет релиз. При этом, если релиз находится в статусе **DRAFT**, то релиз удаляется полностью. Если релиз был выпущен, то на выпущенные платформы шлется удаление, а релиз переходит в статус **TAKE_DOWN.** Если релиз находится в процессе отгрузки, либо уже удален, его удалить нельзя.

Отправка релиза на модерацию

Метод: *POST*
URL: */releases/RELEASE_ID/status/moderate*

Отправляет релиз на модерацию. Метод вернет статус 201, если ошибок в релизе нет и он успешно отправился на модерацию. Релиз нельзя отправить на модерацию, если контракт клиента не подписан.

Получение данных о релизе

Получение данных одного релиза

Метод: *GET*
URL: */releases/RELEASE_ID*
Querystring: *{showOriginalRelease?: boolean}*
Response:

```
{
 releaseId: number;
 articleLabel?: string;
```

```
  internalId?: string;
  isReleased: boolean;
  title: string;
  releaseType: ReleaseType;
  genre: {genreId: number; subGenreId?: number};
  additionalGenres: {genreId: number; subGenreId?: number}[];
  artists: {id: number; role: 'MAIN' | 'ADVANCED' | 'FEAT'}[];
  label: string | null;
  ownEan: string | null;
  contractSigned: boolean;
  language: number | null;
  releaseDate: Date | string | null;
  originalReleaseDate: Date | string | null;
  createDate: string;
  status: ReleaseStatus;
  checklist: ChecklistContainer;
  tracks: TrackInfo[];
  media: {id: number; url: string; filename: string; type: string;}[];
  clineYear: number;
  clineValue: string;
  plineYear: number;
  plineValue: string;
  releaseVersion: string | null;
  streamingPlatforms: number[];
}
```

Возвращает информацию о релизе с его треками

Статусы релизов:

DRAFT – созданный релиз, который еще не был отгружен и не модерировался

EDIT – релиз, в котором изменялись данные после отгрузки либо после модерации. При этом в самом релизе передаются изначальные данные релиза, а в параметре **checklist** передаются изменения, которые еще не были подтверждены модерацией.

ERROR – в релизе присутствует ошибка, которую выявила модерация или которая появилась после отгрузки на платформы. Текст ошибки можно найти в параметре **checklist**.

MODERATE – релиз находится в модерации, при этом если изменить данные в релизе он перейдет в статус EDIT.

WAITING – релиз ожидает отгрузки после модерации.

PARTIAL_UPLOAD – релиз был частично успешно отправлен на платформы.

RELEASED – релиз выпущен на платформах.

TASK – релиз стоит в очереди на отгрузку.

WORKING – релиз отгружается на платформы.

UPLOADED – релиз отгружен на платформы, но дата релиза еще не наступила.

REMOVED – релиз удален с платформ.

Тип ChecklistContainer:

```
{
 data: {field: string; oldValue: any; newValue: any; isAccepted: boolean; description?: string;}[];
 status: 'NEW' | 'PROCESSED';
}
```

Чеклист содержит в себе информацию по изменению значений полей релиза и треков (у каждого трека свой чеклист), при этом если после модерации релиз возвращен в ошибку, то поле **description** будет содержать текст ошибки от модерации. Если релиз уже выпущен, то все изменяемые значения будут попадать в чеклист, а сохранятся в релизе только после одобрения модерации.

Тип TrackInfo

```
{
 trackId: number;
 artists: {id: number; role: 'MAIN' | 'ADVANCED' | 'FEAT'}[];
 persons: {id: number; role: 'MUSIC_AUTHOR' | 'LYRICS_AUTHOR' | 'DIRECTOR'}[];
 genre: {genreId: number; subGenreId?: number};
 additionalGenres: {genreId: number; subGenreId?: number}[];
 language: number | null;
 copyrights: number;
 related: number;
 media: {id: number; url: string; filename: string; type: string;}[];
 ownIsrc: string | null;
 instrumental: boolean;
 adult: boolean;
 title: string;
 trackVersion: string | null;
 text: string | null;
 startPoint: number;
 karaoke?: {rows: {begin: number; end: number; text: string; _blank: boolean;}[]};
 checklist: ChecklistContainer;
 dtCreate: Date | string;
 trackData?: {duration: number;};
 recordingYear?: number;
 clipPreviewFrame: number | null;
 processStatus: 'error' | 'completed';
 processMessage?: string;
 internalId?: string;
}
```

Поиск по релизам

Метод: *POST*
URL: */releases*
Data:

```
{
 search?: string;
 ean?: string;
 releaseId?: number;
```

startDate?: string;
endDate?: string;
startDateRelease?: string;
endDateRelease?: string;
status?: ReleaseStatus;
isrc?: string;
label?: string;
releaseType?: string;
artist?: string;
artistId?: number;
sortBy?: 'status' | 'dt_release' | 'dt_create' | 'nm_release';
sortOrder?: 'asc' | 'desc';
}
Response: *{count: number; data: ReleaseInfo[]}*

Производит поиск по релизам по разным полям. В поле **search** можно указать поисковую строку, поиск будет выполнен по имени релиза, имени артиста, названиям трека и тд.

Получение информации по статусам отгрузки на платформы

Метод: *GET*
URL: */ddex/release/RELEASE_ID*
Response:
{
streamingPlatformId: number;
status: 'UPLOADED' | 'UPDATED' | 'REMOVED' | 'ERROR' | 'WAITING_FOR_DELIVERY';
}[]

Метод возвращает статусы отгрузки по платформам:

UPLOADED – релиз отгружен на платформу.

UPDATED – релиз обновлен на платформе.

REMOVED – релиз удален с платформы.

ERROR – произошла ошибка при отгрузке на платформу.

WAITING_FOR_DELIVERY – релиз отправлен на платформу, ожидается ответ от платформы об успешности отгрузки.

Статистика прослушиваний

Получение значений фильтров и аггрегаций

Метод: *GET*
URL: */statistics/filters*
Response:
{
platformIds: number[];
countryIds: number[];
}

Метод возвращает доступные значения платформ и стран для аггрегаций и фильтрам по статистике. Возвращаются только актуальные значения, по которым есть статистика.

Значения из поля *platformIds* необходимо использовать для полей ***'id_m_list_streaming_platform'***. Значения из поля *countryIds* необходимо использовать для полей *'**id_s_list_country**'*

Получение статистики

Метод: *POST*
URL: */statistics*
Data:
{
dates: {from: string; to: string;}[];
filters: {field: 'id_m_list_streaming_platform' | 'id_m_track' | 'id_s_list_country' | 'id_m_release' | 'nm_label' | 'ext_ean' | 'ext_isrc' | 'artist'; values: string | number;}[]
aggs: {field: 'id_m_list_streaming_platform' | 'id_m_track' | 'id_s_list_country' | 'dt_listen' | 'id_m_release'; format: 'day' | 'week' | 'month' | 'quarter' | 'year'}[];
}
Reponse:
{
count: number;
['id_m_list_streaming_platform' | 'id_m_track' | 'nm_track' | 'isrc' | 'id_s_list_country' | 'code_country' | 'dt_listen' | 'id_m_release' | 'nm_release']: string;
}[];

Платформа позволяет получить полную статистику прослушивания по трекам, релизам, платформам, странам за выбранный период.

В массиве ***dates*** передается фильтр по интересующим датам в формате *YYYY-MM-DD,* при этом можно передать несколько различных временных отрезка, статистика будет расчитана по ним.

В массиве ***filters*** можно передать фильтры для получения статистики только по определенным платформам, трекам, странам, релизам, лейблам, EAN-ам, ISRC либо основному артисту релиза. Значений можно передать несколько, например получить статистику только по определенной платформе в определенной стране.

В массиве ***aggs*** передается список аггрегаций, по которым необходимо сгруппировать данные. Поле *format* используется только при *field=dt_listen* и позволяет сгруппировать статистику по дням, неделям, кварталам, годам.

Пример запроса статистики, сгруппированной по платформам и дате прослушивания:

{"aggs":[{"field":"id_m_list_streaming_platform"},{"field":"dt_listen"}],"filters":[],"dates":[{"from":"2023-01-27","to":"2023-07-27"}]}

Пример ответа:

{data: [{count: 1882; id_m_list_streaming_platform: 291; dt_listen: "2023-01-27 00:00:00"}]}