Yandex Music API

сент. 07, 2021

Оглавление

Глава 1

Yandex.Music.API

1.1 Сборка и тестирование

1.1.1 Сборка

Проекты нацелены на .Net Core 3.1. Сборка вне Visual Studio 2019 производится стандартной командой

```
dotnet build
```

1.1.2 Тестирование

В решение добавлен проект с тестами. Для тестов используется фреймворк xUnit с расширениями FluentAssertions и Xunit.Extensions.Ordering. Т.к. при тестировании используются зависимые данные, все тесты запускаются последовательно и учитывают результат предыдущих необходимых операций.

Для тестирования необходимо добавить в директорию сборки файл appSettings.json со следующим содержимым:

```
{
   "login": "",
   "password": "",
   "token": ""
}
```

Поля объекта соответственно указывают на логин, пароль и токен для доступа к Яндекс.Музыка. При отсутствии токена будет произведена авторизация и получение токена. Для работы с библиотекой рекомендуется использовать способ доступа именно через токен.

Вне Visual Studio 2019 запуск тестов производится стандартной командой

```
dotnet test
```

В результате теста в выходной директории будет создана папка responses, в которой в случае проблем с десериализацией ответов будут сохранены копии ответа, а также создан файл с логом, содержащий описание проблемы. Их можно прикладывать к issues.

1.2 Вспомогательные классы

Для взаимодействия с функциями API используется несколько вспомогательных классов, хранящих в себе информацию о пользователе, а также настройки выполнения.

1.2.1 AuthStorage

Для взаимодействия с функциями API используется несколько вспомогательных классов, хранящих в себе информацию о пользователе, а также настройки выполнения.

Свойства

Context Контекст выполнения http-запросов.

Type: HttpContext

Debug Настройки режима отладки.

Type: DebugSettings

IsAuthorized Флаг авторизации.

Type: bool

Token Токен авторизации.

Type: string

User Информация о пользовательском аккаунте.

Type: YAccount

Методы

```
public AuthStorage(DebugSettings settings = null)
```

Конструктор.

```
public void SetProxy(IWebProxy proxy)
```

Установка прокси.

1.2.2 DebugSettings

Класс с настройками режима отладки.

Свойства

ClearDirectory Флаг очистки директорию с отладочными данными.

Type: bool

LogFileName Имя файла лога, сохраняемого в директории отладки.

Type: string

OutputDir Имя директории для сохранения отладочных данных внутри директории приложения.

Type: string

Методы

```
public DebugSettings(string outputDir, string logFile)
```

Конструктор.

```
public void Clear()
```

Очистка данных.

```
public T Deserialize<T>(string url, string json, JsonSerializerSettings settings)
```

Отладочная функция десериализации объектов.

1.3 YandexMusicAPI

Основной класс для работы с API Яндекс.Музыки. Для удобства разделён на отдельные ветки.

1.3.1 Свойства:

Album API для взаимодействия с альбомами.

Artist API для взаимодействия с артистами.

Library API для взаимодействия с библиотекой.

Playlist API для взаимодействия с плейлистами.

Radio API для взаимодействия с радио.

Search API для взаимодействия с поиском.

Track API для взаимодействия с треками.

User API для взаимодействия с пользовательскими данными.

YAlbumAPI

Методы

```
public async Task<YResponse<YAlbum>> GetAsync(AuthStorage storage, string albumId)
```

Получение альбома в асинхронном режиме.

```
public YResponse<YAlbum> Get(AuthStorage storage, string albumId)
```

Получение альбома.

YArtistAPI

Методы

```
public async Task<YResponse<YArtistBriefInfo>> GetAsync(AuthStorage storage, string artistId)
```

Получение исполнителя в асинхронном режиме.

```
public YResponse<YArtistBriefInfo> Get(AuthStorage storage, string artistId)
```

Получение исполнителя.

YLibraryAPI

Методы

```
public async Task<YResponse<YLibraryTracks>> GetLikedTracksAsync(AuthStorage storage)
```

Получение списка лайкнутых треков в асинхронном режиме.

```
public YResponse<YLibraryTracks> GetLikedTracks(AuthStorage storage)
```

Получение списка лайкнутых треков.

```
public async Task<YResponse<List<YLibraryAlbum>>> GetLikedAlbumsAsync(AuthStorage storage)
```

Получение списка лайкнутых альбомов в асинхронном режиме.

```
public YResponse<List<YLibraryAlbum>> GetLikedAlbums(AuthStorage storage)
```

Получение списка лайкнутых альбомов.

```
public async Task<YResponse<List<YArtist>>> GetLikedArtistsAsync(AuthStorage storage)
```

Получение списка лайкнутых исполнителей в асинхронном режиме.

```
public YResponse<List<YArtist>> GetLikedArtists(AuthStorage storage)
```

Получение списка лайкнутых исполнителей.

```
public async Task<YResponse<List<YLibraryPlaylists>>> GetLikedPlaylistsAsync(AuthStorage storage)
```

Получение списка лайкнутых плейлистов в асинхронном режиме.

```
public YResponse<List<YLibraryPlaylists>> GetLikedPlaylists(AuthStorage storage)
```

Получение списка лайкнутых плейлистов.

```
public async Task<YResponse<YLibraryTrack>> GetDislikedTracksAsync(AuthStorage storage)
```

Получение списка дизлайкнутых треков в асинхронном режиме.

```
public YResponse<YLibraryTrack> GetDislikedTracks(AuthStorage storage)
```

Получение списка дизлайкнутых треков.

```
public async Task<YResponse<YPlaylist>> AddTrackLikeAsync(AuthStorage storage, YTrack track)
```

Добавление трека в список лайкнутых в асинхронном режиме.

```
public YResponse<YPlaylist> AddTrackLike(AuthStorage storage, YTrack track)
```

Добавление трека в список лайкнутых.

```
public async Task<YResponse<YRevision>> RemoveTrackLikeAsync(AuthStorage storage, YTrack track)
```

Удаление трека из списка лайкнутых в асинхронном режиме.

```
public YResponse<YRevision> RemoveTrackLike(AuthStorage storage, YTrack track)
```

Удаление трека из списка лайкнутых.

```
public async Task<YResponse<YRevision>> AddTrackDislikeAsync(AuthStorage storage, YTrack track)
```

Добавление трека в список дизлайкнутых в асинхронном режиме.

```
public YResponse<YRevision> AddTrackDislike(AuthStorage storage, YTrack track)
```

Добавление трека в список дизлайкнутых.

```
public async Task<YResponse<int>> RemoveTrackDislikeAsync(AuthStorage storage, YTrack track)
```

Удаление трека из списка дизлайкнутых в асинхронном режиме.

```
public YResponse<YRevision> RemoveTrackDislike(AuthStorage storage, YTrack track)
```

Удаление трека из списка дизлайкнутых.

```
public async Task<YResponse<string>> AddAlbumLikeAsync(AuthStorage storage, YAlbum album)
```

Добавление альбома в список лайкнутых в асинхронном режиме.

```
public YResponse<string> AddAlbumLike(AuthStorage storage, YAlbum album)
```

Добавление альбома в список лайкнутых.

```
public async Task<YResponse<string>> RemoveAlbumLikeAsync(AuthStorage storage, YAlbum album)
```

Удаление альбома из списка лайкнутых в асинхронном режиме.

```
public YResponse<string> RemoveAlbumLike(AuthStorage storage, YAlbum album)
```

Удаление альбома из списка лайкнутых.

```
public async Task<YResponse<string>> AddArtistLikeAsync(AuthStorage storage, YArtist artist)
```

Добавление исполнителя в список лайкнутых в асинхронном режиме.

```
public YResponse<string> AddArtistLike(AuthStorage storage, YArtist artist)
```

Добавление исполнителя в список лайкнутых.

```
public async Task<YResponse<string>> RemoveArtistLikeAsync(AuthStorage storage, YArtist artist)
```

Удаление исполнителя из списка лайкнутых в асинхронном режиме.

```
public YResponse<string> RemoveArtistLike(AuthStorage storage, YArtist artist)
```

Удаление исполнителя из списка лайкнутых.

```
public async Task<YResponse<string>> AddPlaylistLikeAsync(AuthStorage storage, YPlaylist playlist)
```

Добавление плейлиста в список лайкнутых в асинхронном режиме.

```
public YResponse<string> AddPlaylistLike(AuthStorage storage, YPlaylist playlist)
```

Добавление плейлиста в список лайкнутых.

```
public async Task<YResponse<string>> RemovePlaylistLikeAsync(AuthStorage storage, YPlaylist␣
↪playlist)
```

Удаление плейлиста из списка лайкнутых в асинхронном режиме.

```
public YResponse<string> RemovePlaylistLike(AuthStorage storage, YPlaylist playlist)
```

Удаление плейлиста из списка лайкнутых.

Playlist API

Методы

```
public async Task<YResponse<YLanding>> LandingAsync(AuthStorage storage)
```

Получение списка персональных плейлистов в асинхронном режиме.

```
public YResponse<YLanding> Landing(AuthStorage storage)
```

Получение списка персональных плейлистов.

```
public async Task<YResponse<List<YPlaylist>>> FavoritesAsync(AuthStorage storage)
```

Получение списка избранных плейлистов в асинхронном режиме.

```
public YResponse<List<YPlaylist>> Favorites(AuthStorage storage)
```

Получение списка избранных плейлистов.

```
public async Task<YResponse<YPlaylist>> OfTheDayAsync(AuthStorage storage)
```

Получение плейлиста дня в асинхронном режиме.

```
public YResponse<YPlaylist> OfTheDay(AuthStorage storage)
```

Получение плейлиста дня.

```
public async Task<YResponse<YPlaylist>> DejaVuAsync(AuthStorage storage)
```

Получение плейлиста Дежавю в асинхронном режиме.

```
public YResponse<YPlaylist> DejaVu(AuthStorage storage)
```

Получение плейлиста Дежавю.

```
public async Task<YResponse<YPlaylist>> PremiereAsync(AuthStorage storage)
```

Получение плейлиста Премьера в асинхронном режиме.

```
public YResponse<YPlaylist> Premiere(AuthStorage storage)
```

Получение плейлиста Премьера.

```
public async Task<YResponse<YPlaylist>> MissedAsync(AuthStorage storage)
```

Получение плейлиста Тайник в асинхронном режиме.

```
public YResponse<YPlaylist> Missed(AuthStorage storage)
```

Получение плейлиста Тайник.

```
public async Task<YResponse<YPlaylist>> AliceAsync(AuthStorage storage)
```

Получение плейлиста Алисы в асинхронном режиме.

```
public YResponse<YPlaylist> Alice(AuthStorage storage)
```

Получение плейлиста Алисы.

```
public async Task<YResponse<YPlaylist>> PodcastsAsync(AuthStorage storage)
```

Получение плейлиста Подкасты в асинхронном режиме.

```
public YResponse<YPlaylist> Podcasts(AuthStorage storage)
```

Получение плейлиста Подкасты.

```
public async Task<YResponse<YPlaylist>> RewindAsync(AuthStorage storage)
```

Получение плейлиста Мой 2020 в асинхронном режиме.

```
public YResponse<YPlaylist> Rewind(AuthStorage storage)
```

Получение плейлиста Мой 2020.

```
public async Task<YResponse<YPlaylist>> GetAsync(AuthStorage storage, string user, string kinds)
```

Получение плейлиста в асинхронном режиме.

```
public YResponse<YPlaylist> Get(AuthStorage storage, string user, string kinds)
```

Получение плейлиста.

```
public async Task<YResponse<YPlaylist>> GetAsync(AuthStorage storage, YPlaylist playlist)
```

Получение плейлиста в асинхронном режиме.

```
public YResponse<YPlaylist> Get(AuthStorage storage, YPlaylist playlist)
```

Получение плейлиста.

```
public async Task<YResponse<YPlaylist>> CreateAsync(AuthStorage storage, string name)
```

Создание плейлиста в асинхронном режиме.

Примечание: Следующие операции можно выполнять только над собственными плейлистами

```
public YResponse<YPlaylist> Create(AuthStorage storage, string name)
```

Создание плейлиста.

```
public async Task<YResponse<YPlaylist>> RenameAsync(AuthStorage storage, string kinds, string name)
```

Переименование плейлиста в асинхронном режиме.

```
public YResponse<YPlaylist> Rename(AuthStorage storage, string kinds, string name)
```

Переименование плейлиста.

```
public Task<YResponse<YPlaylist>> RenameAsync(AuthStorage storage, YPlaylist playlist, string name)
```

Переименование плейлиста в асинхронном режиме.

```
public YResponse<YPlaylist> Rename(AuthStorage storage, YPlaylist playlist, string name)
```

Переименование плейлиста.

```
public async Task<bool> DeleteAsync(AuthStorage storage, string kinds)
```

Удаление плейлиста в асинхронном режиме.

```
public bool Delete(AuthStorage storage, string kinds)
```

Удаление плейлиста.

```
public Task<bool> DeleteAsync(AuthStorage storage, YPlaylist playlist)
```

Удаление плейлиста в асинхронном режиме.

```
public bool Delete(AuthStorage storage, YPlaylist playlist)
```

Удаление плейлиста.

```
public async Task<YResponse<YPlaylist>> InsertTracksAsync(AuthStorage storage, YPlaylist playlist,␣
↪params YTrack[] tracks)
```

Добавление треков в асинхронном режиме.

```
public YResponse<YPlaylist> InsertTracks(AuthStorage storage, YPlaylist playlist, params YTrack[]␣
↪tracks)
```

Добавление треков.

```
public async Task<YResponse<YPlaylist>> DeleteTracksAsync(AuthStorage storage, YPlaylist playlist,␣
↪params YTrack[] tracks)
```

Удаление треков в асинхронном режиме.

```
public YResponse<YPlaylist> DeleteTracks(AuthStorage storage, YPlaylist playlist, params YTrack[]␣
↪tracks)
```

Удаление треков.

Radio API

Методы

```
public async Task<YResponse<YStationsDashboard>> GetStationsDashboardAsync(AuthStorage storage)
```

Получение списка рекомендованных радиостанций в асинхронном режиме.

```
public YResponse<YStationsDashboard> GetStationsDashboard(AuthStorage storage)
```

Получение списка рекомендованных радиостанций.

```
public async Task<YResponse<List<YStation>>> GetStationsAsync(AuthStorage storage)
```

Получение списка радиостанций в асинхронном режиме.

```
public YResponse<List<YStation>> GetStations(AuthStorage storage)
```

Получение списка радиостанций.

```
public async Task<YResponse<List<YStation>>> GetStationAsync(AuthStorage storage, string type,␣
↪string tag)
```

Получение информации о радиостанции в асинхронном режиме.

```
public YResponse<List<YStation>> GetStation(AuthStorage storage, string type, string tag)
```

Получение информации о радиостанции.

```
public Task<YResponse<List<YStation>>> GetStationAsync(AuthStorage storage, YStationId id)
```

Получение информации о радиостанции в асинхронном режиме.

```
public YResponse<List<YStation>> GetStation(AuthStorage storage, YStationId id)
```

Получение информации о радиостанции.

```
public async Task<YResponse<YStationSequence>> GetStationTracksAsync(AuthStorage storage, YStation␣
↪station, string prevTrackId = "")
```

Получение треков радиостанции в асинхронном режиме.

```
public YResponse<YStationSequence> GetStationTracks(AuthStorage storage, YStation station, string␣
↪prevTrackId = "")
```

Получение треков радиостанции.

```
public async Task<YResponse<string>> SetStationSettings2Async(AuthStorage storage, YStation␣
↪station, YStationSettings2 settings)
```

Установка настроек радиостанции в асинхронном режиме.

```
public YResponse<string> SetStationSettings2(AuthStorage storage, YStation station,␣
↪YStationSettings2 settings)
```

Установка настроек радиостанции.

Search API

Методы

```
public async Task<YResponse<YSearch>> TrackAsync(AuthStorage storage, string trackName, int␣
↪pageNumber = 0)
```

Поиск по трекам в асинхронном режиме.

```
public YResponse<YSearch> Track(AuthStorage storage, string trackName, int pageNumber = 0)
```

Поиск по трекам.

```
public async Task<YResponse<YSearch>> AlbumsAsync(AuthStorage storage, string albumName, int␣
↪pageNumber = 0)
```

Поиск по альбомам в асинхронном режиме.

```
public YResponse<YSearch> Albums(AuthStorage storage, string albumName, int pageNumber = 0)
```

Поиск по альбомам.

```
public async Task<YResponse<YSearch>> ArtistAsync(AuthStorage storage, string artistName, int␣
↪pageNumber = 0)
```

Поиск по исполнителям в асинхронном режиме.

```
public YResponse<YSearch> Artist(AuthStorage storage, string artistName, int pageNumber = 0)
```

Поиск по исполнителям.

```
public async Task<YResponse<YSearch>> PlaylistAsync(AuthStorage storage, string playlistName, int␣
↪pageNumber = 0)
```

Поиск по плейлистам в асинхронном режиме.

```
public YResponse<YSearch> Playlist(AuthStorage storage, string playlistName, int pageNumber = 0)
```

Поиск по плейлистам.

```
public async Task<YResponse<YSearch>> VideosAsync(AuthStorage storage, string videoName, int␣
↪pageNumber = 0)
```

Поиск по видеозаписям в асинхронном режиме.

```
public YResponse<YSearch> Videos(AuthStorage storage, string videoName, int pageNumber = 0)
```

Поиск по видеозаписям.

```
public async Task<YResponse<YSearch>> UsersAsync(AuthStorage storage, string userName, int␣
↪pageNumber = 0)
```

Поиск по пользователям в асинхронном режиме.

```
public YResponse<YSearch> Users(AuthStorage storage, string userName, int pageNumber = 0)
```

Поиск по пользователям.

```
public async Task<YResponse<YSearch>> SearchAsync(AuthStorage storage, string searchText,␣
↪YSearchType searchType, int page = 0)
```

Поиск в асинхронном режиме.

```
public YResponse<YSearch> Search(AuthStorage storage, string searchText, YSearchType searchType,␣
↪int page = 0)
```

Поиск.

```
public async Task<YResponse<YSearchSuggest>> SuggestAsync(AuthStorage storage, string searchText)
```

Получение подсказок по поиску в асинхронном режиме.

```
public YResponse<YSearchSuggest> Suggest(AuthStorage storage, string searchText)
```

Получение подсказок по поиску.

Track API

Методы

```
public async Task<YResponse<List<YTrack>>> GetAsync(AuthStorage storage, string trackId)
```

Получение трека в асинхронном режиме.

```
public YResponse<List<YTrack>> Get(AuthStorage storage, string trackId)
```

Получение трека.

Примечание: Здесь и далее trackKey формируется в формате «<id альбома>:<id трека>».

```
public async Task<YResponse<List<YTrackDownloadInfo>>> GetMetadataForDownloadAsync(AuthStorage␣
↪storage, string trackKey, bool direct)
```

Получение метаданных для загрузки в асинхронном режиме.

```
public YResponse<List<YTrackDownloadInfo>> GetMetadataForDownload(AuthStorage storage, string␣
↪trackKey, bool direct = false)
```

Получение метаданных для загрузки.

```
public async Task<YResponse<List<YTrackDownloadInfo>>> GetMetadataForDownloadAsync(AuthStorage␣
↪storage, YTrack track, bool direct = false)
```

Получение метаданных для загрузки в асинхронном режиме.

```
public YResponse<List<YTrackDownloadInfo>> GetMetadataForDownload(AuthStorage storage, YTrack␣
↪track, bool direct = false)
```

Получение метаданных для загрузки.

```
public async Task<YStorageDownloadFile> GetDownloadFileInfoAsync(AuthStorage storage,␣
↪YTrackDownloadInfo metadataInfo)
```

Получение данных для формирования ссылки в асинхронном режиме.

```
public YStorageDownloadFile GetDownloadFileInfo(AuthStorage storage, YTrackDownloadInfo␣
↪metadataInfo)
```

Получение данных для формирования ссылки.

```
public string GetFileLink(AuthStorage storage, string trackKey)
```

Получение ссылки.

```
public string GetFileLink(AuthStorage storage, YTrack track)
```

Получение ссылки.

```
public void ExtractToFile(AuthStorage storage, string trackKey, string filePath)
```

Сохранение в файл.

```
public void ExtractToFile(AuthStorage storage, YTrack track, string filePath)
```

Сохранение в файл.

```
public byte[] ExtractData(AuthStorage storage, string trackKey)
```

Получение данных в виде двоичного массива.

```
public byte[] ExtractData(AuthStorage storage, YTrack track)
```

Получение данных в виде двоичного массива.

YUserAPI

Методы

```
public async Task AuthorizeAsync(AuthStorage storage, string login, string password)
```

Авторизация в асинхронном режиме с использованием логина и пароля.

```
public void Authorize(AuthStorage storage, string login, string password)
```

Авторизация с использованием логина и пароля.

Совет: Рекомендуется использовать эти методы только для первоначального получения токена, а в дальнейшем использовать его.

```
public async Task AuthorizeAsync(AuthStorage storage, string token)
```

Авторизация в асинхронном режиме с использованием токена.

```
public void Authorize(AuthStorage storage, string token)
```

Авторизация с использованием токена.

Предупреждение: Необходимо обязательно выполнить авторизацию перед использованием функционала API.

```
public async Task<YResponse<YAccountResult>> GetUserAuthAsync(AuthStorage storage)
```

Получение информации об авторизации в асинхронном режиме.

```
public YResponse<YAccountResult> GetUserAuth(AuthStorage storage)
```

Получение информации об авторизации.

Глава 2

Yandex.Music.Client

2.1 YandexMusicClient

Класс клиента для работы с API Яндекс.Музыки. Реализует основной функционал получения объектов для взаимодействия с API. Для самих объектов функционал реализован в виде методов-расширений.

2.1.1 Свойства

Context Account, с которым работает клиент.

Type: YAccount

IsAuthorized Флаг авторизации

Type: bool

2.1.2 Методы

```
public YandexMusicClient(DebugSettings settings = null)
```

Конструктор.

```
public bool Authorize(string login, string password)
```

Авторизация с использованием логина и пароля.

```
public bool Authorize(string token)
```

Авторизация с использованием токена.

```
public YTrack GetTrack(string id)
```

Получение трека по идентификатору.

```
public YAlbum GetAlbum(string id)
```

Получение альбома по идентификатору.

```
public YArtistBriefInfo GetArtist(string id)
```

Получение исполнителя по идентификатору.

```
public YPlaylist GetPlaylist(string user, string id)
```

Получение плейлиста по пользователю и идентификатору.

```
public List<YPlaylist> GetPersonalPlaylists()
```

Получение персональных плейлистов.

```
public List<YPlaylist> GetFavorites()
```

Получение списка избранных плейлистов.

```
public YPlaylist GetAlice()
```

Получение плейлиста Алисы.

```
public YPlaylist GetDejaVu()
```

Получение плейлиста Дежавю.

```
public YPlaylist GetMissed()
```

Получение плейлиста Тайник.

```
public YPlaylist GetOfTheDay()
```

Получение плейлиста дня.

```
public YPlaylist GetPodcasts()
```

Получение плейлиста Подкасты.

```
public YPlaylist GetRewind()
```

Получение плейлиста Мой 2020.

```
public YPlaylist GetPremiere()
```

Получение плейлиста Премьера.

```
public YPlaylist CreatePlaylist(string name)
```

Создание плейлиста.

```
public YSearch Search(string searchText, YSearchType searchType, int page = 0)
```

Поиск.

```
public YSearchSuggest GetSearchSuggestions(string searchText)
```

Подсказки по поиску.

```
public List<YStation> GetRadioDashboard()
```

Получение списка рекомендованных радиостанций.

```
public List<YStation> GetRadioStations()
```

Получение списка радиостанций.

```
public YStation GetRadioStation(YStationId id)
```

Получение радиостанции по идентификатору.