FTC-Music-Player Documentation

Contents

Namespace `FTC-Music-Player`

Sub-modules

- FTC-Music-Player.Home page
- FTC-Music-Player.Test Files
- FTC-Music-Player.api_client
- FTC-Music-Player.data_models
- FTC-Music-Player.lib
- FTC-Music-Player.localfiles
- FTC-Music-Player.search_models

Module `FTC-Music-Player.Home page`

Functions

Function `show_value`

```
def show_value(
    selected_option
)
```

Function `temp_text`

```
def temp_text(
    e
)
```

Namespace `FTC-Music-Player.Test Files`

Sub-modules

- FTC-Music-Player.Test Files.hometest

Module `FTC-Music-Player.Test Files.hometest`

Namespace `FTC-Music-Player.api_client`

Sub-modules

- FTC-Music-Player.api_client.Youtube
- FTC-Music-Player.api_client.api_client

Namespace `FTC-Music-Player.api_client.Youtube`

Sub-modules

- FTC-Music-Player.api_client.Youtube.example
- FTC-Music-Player.api_client.Youtube.models
- FTC-Music-Player.api_client.Youtube.youtube

Module `FTC-Music-Player.api_client.Youtube.example`

Module `FTC-Music-Player.api_client.Youtube.models`

Classes

Class `GetAlbumSongsResponse`

```
class GetAlbumSongsResponse(
    has_error: bool,
    error: str,
    songs: list[FTC-Music-Player.api_client.Youtube.models.OnlineSong]
)
```

Class variables

Variable `error` Type: `str`

Variable `has_error` Type: `bool`

Variable `songs` Type: `list[FTC-Music-Player.api_client.Youtube.models.OnlineSong]`

Class `GetArtistAlbumsResponse`

```
class GetArtistAlbumsResponse(
    has_error: bool,
    error: str,
    albums: list[FTC-Music-Player.api_client.Youtube.models.OnlineAlbum]
)
```

Class variables

Variable `albums` Type: `list[FTC-Music-Player.api_client.Youtube.models.OnlineAlbum]`

Variable `error` Type: `str`

Variable `has_error` Type: `bool`

Class `GetArtistSongsResponse`

```
class GetArtistSongsResponse(
    has_error: bool,
    error: str,
    songs: list[FTC-Music-Player.api_client.Youtube.models.OnlineSong]
)
```

Class variables

Variable `error` Type: `str`

Variable `has_error` Type: `bool`

Variable `songs` Type: `list[FTC-Music-Player.api_client.Youtube.models.OnlineSong]`

Class `GetSongUrlResponse`

```
class GetSongUrlResponse(
    has_error: bool,
    error: str,
    url: str
)
```

Class variables

Variable `error` Type: `str`

Variable `has_error` Type: `bool`

Variable `url` Type: `str`

Class `OnlineAlbum`

```
class OnlineAlbum(
    id: str,
    artist_id: str,
    name: str,
    cover_art: str,
    songs: list[FTC-Music-Player.api_client.Youtube.models.OnlineSong]
)
```

Immutable collection of Songs made by the same Artist and grouped togther.

- name – Name of the Album.
- arist – Name of the Artist that made the Album.
- songs – List of Songs in the Album.
- _path – URL of the playlist on YT.

Ancestors (in MRO)

- data_models.Album
- data_models.Content

Class variables

Variable `artist_id` Type: `str`

Variable `cover_art` Type: `str`

Variable `id` Type: `str`

Class `OnlineArtist`

```
class OnlineArtist(
    id: str,
    name: str,
    cover_art: str,
    albums: list[FTC-Music-Player.api_client.Youtube.models.OnlineAlbum],
    songs: list[FTC-Music-Player.api_client.Youtube.models.OnlineSong]
)
```

An Artist class to model the data for an artist (in our case, a YouTube channel).

- name – Name of the artist.

- albums – Albums (playlists) made by the artist.
- songs – Songs (uploads).
- id – Unique identifier.

Ancestors (in MRO)

- data_models.Artist

Class variables

Variable `cover_art` Type: `str`

Variable `id` Type: `str`

Class `OnlineSong`

```
class OnlineSong(
    id: str,
    artist_id: str,
    name: str,
    url: str,
    cover_art: str,
    duration: datetime.timedelta
)
```

Models a single Song that can be a part of zero or more Albums or Playlists, as well as followed by zero or more Users.

- name – Song’s name.

- artist – Creator of the song (channel that uploaded it).

- _path – Either a local path to the song file or a URL generated from the C# back-end.

Ancestors (in MRO)

- data_models.Song
- data_models.Content

Class variables

Variable `artist_id` Type: `str`

Variable `cover_art` Type: `str`

Variable `id` Type: `str`

Class `SearchRequest`

```
class SearchRequest(
    query,
    artist_count,
    album_count,
    song_count
)
```

Class variables

Variable `album_count` Type: `int`

Variable `artist_count` Type: `int`

Variable `query` Type: `str`

Variable `song_count` Type: `int`

Class `SearchResponse`

```
class SearchResponse(
    has_error: bool,
    error: str,
    artists: list[FTC-Music-Player.api_client.Youtube.models.OnlineArtist],
    albums: list[FTC-Music-Player.api_client.Youtube.models.OnlineAlbum],
    songs: list[FTC-Music-Player.api_client.Youtube.models.OnlineSong]
)
```

Class variables

Variable `albums` Type: `list[FTC-Music-Player.api_client.Youtube.models.OnlineAlbum]`

Variable `artists` Type: `list[FTC-Music-Player.api_client.Youtube.models.OnlineArtist]`

Variable `error` Type: `str`

Variable `has_error` Type: `bool`

Variable `songs` Type: `list[FTC-Music-Player.api_client.Youtube.models.OnlineSong]`

Module `FTC-Music-Player.api_client.Youtube.youtube`

This file builds the requests to the API server via the api_client and parses the responses into the models defined in models.py.

Functions

Function `getAlbumSongs`

```
def getAlbumSongs(
    album_id: str
) -> Youtube.models.GetAlbumSongsResponse
```

Invokes a GetAlbumSongs Request via api_client.

Takes in a album_id as a string.

Returns a GetAlbumSongsResponse containing the songs of the album.

Function `getArtistAlbums`

```
def getArtistAlbums(
    artist_id: str
) -> Youtube.models.GetArtistAlbumsResponse
```

Invokes a GetArtistAlbums Request via api_client.

Takes in a artist_id as a string.

Returns a GetArtistAlbumsResponse containing the albums of the artist.

Function `getArtistSongs`

```
def getArtistSongs(
    artist_id: str
) -> Youtube.models.GetArtistSongsResponse
```

Invokes a GetArtistSongs Request via api_client.

Takes in a artist_id as a string.

Returns a GetArtistSongsResponse containing the songs of the artist.

Function `getSongUrl`

```
def getSongUrl(
    song_id: str
) -> Youtube.models.GetSongUrlResponse
```

Invokes a GetSongUrl Request via api_client.

Takes in a song_id as a string.

Returns a GetSongUrlRespose containing the url of the song.

Function `search`

```
def search(
    request: Youtube.models.SearchRequest
) -> Youtube.models.SearchResponse
```

Invokes a Search Request via api_client.

Takes in a SearchRequest object.

Returns a SearchResponse object containing the results of the search request.

Module `FTC-Music-Player.api_client.api_client`

This file directly interatcs with the API server via “sendApiRequest” function which takes in the controller, request, and params as strings, sends a request to the server and returns the response (in json format).

It also contains the ApiControllers and ApiRequests enums which are used to build the request url as well as the serverIp variable which is used to build the request url.

Functions

Function `sendApiRequest`

```
def sendApiRequest(
    controller: str,
    request: str,
    params: str
) -> str
```

Classes

Class `ApiControllers`

```
class ApiControllers(
    value,
    names=None,
    *,
    module=None,
    qualname=None,
    type=None,
    start=1
)
```

An enumeration.

Ancestors (in MRO)

- enum.Enum

Class variables

Variable `Youtube`

Class `ApiRequests`

```
class ApiRequests(
    value,
    names=None,
    *,
    module=None,
    qualname=None,
    type=None,
    start=1
)
```

An enumeration.

Ancestors (in MRO)

- enum.Enum

Class variables

Variable `AlbumSongs`

Variable `ArtistAlbums`

Variable `ArtistSongs`

Variable `AudioUrl`

Variable `Search`

Module `FTC-Music-Player.data_models`

Defines several classes modelling data that will be received from the C# backend to be used by the Python frontend:

- User – A class to represent a single User of the application.
- Content – Top level class modelling any kind of content an Artist could make.
- Album : Content – A collection of Songs.
- Playlist : Album – A /mutable/ collection of Songs created by the User.
- Song : Content – A single Song.
- Artist – An entity that has zero or more Content.

Classes

Class `Album`

```
class Album(
    name: str,
    artist,
    songs: list[FTC-Music-Player.data_models.Song],
    path: str
)
```

Immutable collection of Songs made by the same Artist and grouped togther.

- name – Name of the Album.
- arist – Name of the Artist that made the Album.
- songs – List of Songs in the Album.
- _path – URL of the playlist on YT.

Ancestors (in MRO)

- FTC-Music-Player.data_models.Content

Descendants

- FTC-Music-Player.data_models.Playlist

Class variables

Variable `songs` Type: `list[FTC-Music-Player.data_models.Song]`

Class `Artist`

```
class Artist(
    name: str,
    id: str = 'FTC',
    albums: list[FTC-Music-Player.data_models.Album] = [],
    songs: list[FTC-Music-Player.data_models.Song] = []
)
```

An Artist class to model the data for an artist (in our case, a YouTube channel).

- name – Name of the artist.

- albums – Albums (playlists) made by the artist.
- songs – Songs (uploads).
- id – Unique identifier.

Class variables

Variable `albums` Type: `list[FTC-Music-Player.data_models.Album]`

Variable `id` Type: `str`

Variable `name` Type: `str`

Variable `songs` Type: `list[FTC-Music-Player.data_models.Song]`

Class `Content`

```
class Content
```

Class modelling any form of Content made my an Artist, inherited by several other classes.

- artist – Channel that uploaded this content.
- _path – Either a local path to the file or a URL generated from the C# back-end.

Descendants

- FTC-Music-Player.data_models.Album
- FTC-Music-Player.data_models.Song

Class variables

Variable `artist`

Variable `name` Type: `str`

Class `Playlist`

```
class Playlist(
    name: str,
    path: str,
    songs: list[FTC-Music-Player.data_models.Song] = []
)
```

Playlist class to be used in the interface between the back- and front-ends.

- name – Name of the Playlist.
- arist – Name of the Artist that made the Playlist.
- songs – List of Songs in the Playlist.
- _path – Local path to the playlist folder (of the form “./playlist/”).
- add_song() – Adds a Song to the list of Songs.

Ancestors (in MRO)

- FTC-Music-Player.data_models.Album
- FTC-Music-Player.data_models.Content

Methods

Method `add_song`

```
def add_song(
    self,
    song: FTC-Music-Player.data_models.Song
)
```

Class `Song`

```
class Song(
    name: str,
    artist,
    path: str,
    duration: datetime.timedelta
)
```

Models a single Song that can be a part of zero or more Albums or Playlists, as well as followed by zero or more Users.

- name – Song’s name.
- artist – Creator of the song (channel that uploaded it).
- _path – Either a local path to the song file or a URL generated from the C# back-end.

Ancestors (in MRO)

- FTC-Music-Player.data_models.Content

Class variables

Variable `duration` Type: `datetime.timedelta`

Methods

Method `play`

```
def play(
    self
)
```

Start playing the current song from _path

Class `User`

```
class User(
    username: str,
    token: str
)
```

A User class to model the data for a user of the application.

- username – Personal identifier
- token – Unique identifier made by concatenating the user’s username with their password and computing their SHA256.
- playlists – List of playlists/albums the user has saved.
- favourites – Special playlist of the user’s favourite songs.
- followed_artists – List of all the artists that the user has followed.
- followed_albums – List of all the albums that the user has followed.

Class variables

Variable `favourites` Type: `FTC-Music-Player.data_models.Playlist`

Variable `followed_albums` Type: `list[FTC-Music-Player.data_models.Album]`

Variable `followed_artists` Type: `list[FTC-Music-Player.data_models.Artist]`

Variable `playlists` Type: `list[FTC-Music-Player.data_models.Playlist]`

Variable `token` Type: `str`

Variable `username` Type: `str`

Methods

Method `change_username`

```
def change_username(
    self,
    new_username: str
)
```

Method `create_playlist`

```
def create_playlist(
    self,
    name: str
)
```

Method `follow_album`

```
def follow_album(
    self,
    album: FTC-Music-Player.data_models.Album
)
```

Method `follow_artist`

```
def follow_artist(
    self,
    artist: FTC-Music-Player.data_models.Artist
)
```

Method `like_song`

```
def like_song(
    self,
    song: FTC-Music-Player.data_models.Song
)
```

Module `FTC-Music-Player.lib`

Definitions of some generally useful functions to use throughout the project.

Functions

Function `TODO`

```
def TODO(
    s: str
)
```

Function `err`

```
def err(
    name: str,
    e: str
)
```

General function for logging errors.

- name: Name of the calling module/function.
- s : Displayed log.

Function `logger`

```
def logger(
    name: str,
    s: str
)
```

General logging function for use by the library.

- name: Name of the calling module/function.
- s : Displayed log.

Module `FTC-Music-Player.localfiles`

Testing how to select a main music folder and a song to play

Functions

Function `addtoqu`

```
def addtoqu(
    listbox
)
```

Function `getfile`

```
def getfile(
    listbox
)
```

Function `getselected`

```
def getselected(
    event,
    listbox
)
```

Function `quclear`

```
def quclear(
    listbox
)
```

Module `FTC-Music-Player.search_models`

Classes for data sent to and from the C# back-end.

Classes

Class `SearchRequest`

```
class SearchRequest(
    query,
    artist_count,
    album_count,
    song_count
)
```

Class modelling data to be sent to the C# back-end.

Class variables

Variable `album_count` Type: `int`

Variable `artist_count` Type: `int`

Variable `query` Type: `str`

Variable `song_count` Type: `int`

Class `SearchResults`

```
class SearchResults(
    artists: list[data_models.Artist],
    albums: list[data_models.Album],
    songs: list[data_models.Song]
)
```

Class to model data sent back by the C# back-end.

Class variables

Variable `albums` Type: `list[data_models.Album]`

Variable `artists` Type: `list[data_models.Artist]`

Variable `songs` Type: `list[data_models.Song]`

Generated by *pdoc* 0.10.0 (https://pdoc3.github.io).