DeepFilm Data API v5

The DeepFilm Data API v5 allows you to programmatically access the resources used on https://deepfilm.de, which is a JavaScript client and GUI (*Graphical User Interface*) for the Data API (*Application Programming Interface*).

General information

The DeepFilm Data API v5 is available at `deepfilm-v5.runstorageapis.com` and only supports **https** connections and **GET** requests.

To perform certain actions, you need to be authenticated and authorized. This is done by an **API key**, a 64 character string. It can be obtained from the `api_key` cookie after a user has signed in successfully via *auth.runstorage.io*.

The API supports the following methods:

[method]	description
search	This method gets a list of movies, which satisfy certain criteria. It is used for all operations that involve listing movies: e.g. search, advanced search, list, etc.
movie	The movie method is used to obtain details about a public movie or a movie in a user's personal library.
cloud	This method can be used to manage a user's personal library. It supports adding movies, but also changing their respective tags.

To call the API, simply construct a REST request with the respective method and parameters:

```
GET https://deepfilm-v5.runstorageapis.com/[method]?[parameters]
```

If there are any questions, feel free to contact us via https://www.runstorage.com/contact.

search

Get movies matching parameter q.

Parameters:

- `[q = QUERY]`: This contains the search query. The parameter q know three modes:
  - **mode 0**: Mode 0 is active, when the parameter is not present. This tells the API to list movies without filtering.
  - **mode 1**: To use mode 1 (normal search), just send a normal search query as parameter q.
  - **mode 2**: Mode 2 is the advanced search mode. It is used by specifying a double underscore and then the a URL encoded JSON object with the different filters. (See paragraph: Advanced Search Filters)
- `[limit = LIMIT, default: 10]`: The maximal number of movies to be returned by the API in one request (called page).
- `[page = PAGE, default: 1]`: When working with the default value in *limit*, page = 1 will return the first 10 movies. If set to 2 it will return the next 10 etc. The maximum number of pages can be calculated with this formula: `Math.ceil(result.total / limit)` (where *result* is the parsed JSON response of the first request).
- `[order = ORDER, default: date]`: The factor by which to order the API results. By default this is by date (newest first). Other options are *rand* and *time_added*.
- `[key = KEY, default: PUBLIC]`: If this value is specified and is not *PUBLIC*, the search will be performed on the user's library instead of the public list.

Advanced Search Filters:
These are the filters for the advanced search capability. For details on each filter, please read the general documentation of DeepFilm.

Sender => broadcaster; *Normaler Suchbegriff => tags*; *Titel => title*; *Beschreibung => description*; *Sendedatum => date (dd.mm.yyyy)*; *Minimale Länge => min_length (hh:mm or hh:mm:ss)*

Response:

- `movies`: An array of results or *true* if the search was successful but no results could be found.
  - `id`: The movie's unique numeric ID.
  - `broadcaster`: The name of the movie's broadcaster.

    - `title`: The movie's title.
    - `length`: Its length in seconds.
    - `status`: The movie's status. This is *PUBLIC* when the movie is not in a user's account. Otherwise it can be: *QUEUED*, *CANCELED*, *IN_PROGRESS*, *SUCCESS* or *FAILURE*.
    - `tags`: The movie's tags separated by semicolon.
    - `thumbnail`: The filename of the movie's thumbnail if applicable.
- `total`: The total amount of search results.
- `[error]`: An error message. This is only populated if success is false.
- `success`: Whether the request was successful. Should be true.

movie

Get information about a specified movie.

Parameters:

- `id`: The ID of the movie that information should be retrieved of.
- `[key = KEY, default: PUBLIC]`: If this is not *PUBLIC*, but a user's key, the method can, in addition to public movies, also return movies in the respective user's account.

Response:

- `movie`: An object holding the information requested.
    - `id`: The movie's unique numeric ID. The same value as the parameter id sent with the request.
    - `owner`: The numeric user ID of the owner. This is *PUBLIC* if the movie doesn't belong to a user's account.
    - `broadcaster`: The name of the broadcaster.
    - `title`: The movie's title.
    - `topic`: The topic the broadcaster says the movie is about.
    - `length`: The length of the movie in seconds.
    - `date`: An object containing relevant dates about the movie.
        - `begin`: The broadcasting date as a Unix timestamp.
        - `end`: The time the broadcasting ended. This value can be obtained by adding the length to the beginning date.
    - `description`: A description of the movie.
    - `website`: A link to the broadcaster's website with further detail about the movie.

- `url_broadcaster`: A direct link to the video file on the broadcaster's servers. Please don't use this URL to play the movie, instead rely on the *player* object.
- `url_runstorage`: An ID within our [backupci.com]() archives. This value is only populated if the movie is in a user's account and *status* is *SUCCESS*. Please don't use this technical ID to play the movie, instead rely on the *player* object.
- `filesize`: An object containing information about the movie's file size.
  - `filesize`: The movie's real file size in bytes.
  - `available`: A boolean of whether the file size property contains valid information.
  - `estimate`: An estimated file size to be used when no real size is available.
- `status`: The movie's status. This is *PUBLIC* when the movie is not in a user's account. Otherwise it can be one of *QUEUED*, *CANCELED*, *IN_PROGRESS*, *SUCCESS* or *FAILURE*.
- `time_added`: A Unix timestamp of the moment the movie was added to DeepFilm (for public movies) or the moment it was added to the user's library (for user's private movies).
- `tags`: An object holding the movie's tags.
  - `tags`: An array of tags.
  - `string`: The movie's tags separated by semicolon.
- `search`: The movie's tags separated by semicolon with a few special chars removed.
- `thumbnail`: An object containing details about the movie's thumbnail image.
  - `thumbnail`: The thumbnail's filename on our servers.
  - `available`: A boolean whether a thumbnail is available for this movie.
  - `url`: A link to retrieve the thumbnail. This is the preferred method of using the thumbnail.
- `hash`: A validation hash the identifies each movie. It is the same no matter if the movie is public or in a user's library.
- `player`: An object consisting of information to play or embed the movie.
  - `url`: The link that should be used to play the movie. It is automatically created and is the preferred method of playing the movie.
  - `type`: The type of player to use. This can be `https`, `http`, `drive` or `other`.
- `permissions`: An object holding the information which operations can be executed on this movie.
  - `read`: Is always *true*. Otherwise you would have received false in the global property success.
  - `write`: Whether you can update (change tags) the movie.
  - `add`: Whether this movie can be added to a user's library.

- `[error]`: An error message. This is only populated if success is false.
- `success`: Whether the request was successful. Should be true.

cloud

Manage a movie within a user's account.

Parameters:

- `id`: The movie that should be added to the user's account or that should be changed within the user's account.
- `[tags = TAGS, default: ]`: If the parameter tags is specified, we assume that *id* is within the user's account and should be changed to use the tags sent in this parameter. If you want to add *id* to the user's account, do not send this parameter.
- `key`: The user's API key for authentication and authorization.

Response:

- `[id]`: If the movie was added, this is populated with the ID the movie has within the user's account. If a movie within the account was changed, this value does not exist.
- `[error]`: An error message. This is only populated if success is false.
- `success`: Whether the request was successful. Should be true.