
tautulli

Nate Harris

Sep 08, 2023

CONTENTS:

1	Introduction	1
1.1	Motivation	1
2	Documentation	3
2.1	The RawAPI class	3
2.2	The ObjectAPI class	33
2.3	Models	62
2.4	Tools	195
3	Indices and tables	197
	Python Module Index	199
	Index	201

INTRODUCTION

The `tautulli` Python modules allows you to interact with a Tautulli instance's API.

This modules covers nearly 100% of Tautulli's available API endpoints.

The module has built-in type and choice checks to prevent users from making incorrect API requests.

Most functions return either a True/False boolean based on the "result" message from the server, or a raw JSON object parsed from the server's "data" response.

A handful of functions will instead return raw strings or bytearrays (in the case of downloading files)

1.1 Motivation

Myself alone, I've probably written a dozen Python scripts and bots that interact with Tautulli in one way or another. And each time, I have to write out the API endpoints and parse the response data myself.

I figured it was about time someone made a Python library that can handle that part.

DOCUMENTATION

2.1 The RawAPI class

```
class tautulli.api.json_api.RawAPI(base_url, api_key, verbose=False, verify=True, ssl_verify=True)
```

Bases: object

```
activity(session_key=None, session_id=None)
```

Get the current activity on the Plex Media Server

Parameters

- **session_key** (*int, optional*) – Session key for the session info to return
- **session_id** (*str, optional*) – Session ID of the session info to return

Returns

Dict of data

Return type

dict

```
add_newsletter_config(agent_id)
```

Add a new newsletter notification agent

Parameters

agent_id (*int*) – Newsletter type to add

Returns

True if successful, *False* if unsuccessful

Return type

bool

```
add_notifier_config(agent_id)
```

Add a new notifier notification agent

Parameters

agent_id (*int*) – Notification agent to add

Type

int

Returns

True if successful, *False* if unsuccessful

Return type

bool

backup_config()

Backup the config.ini file

Returns

True if successful, *False* if unsuccessful

Return type

bool

backup_database()

Backup the plexpy.db database

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_all_library_history(server_id, section_id, row_ids=None)

Delete all Tautulli history for a specific library

Parameters

- **server_id** (*str*) – Plex server identifier of the library section
- **section_id** (*str*) – ID of the Plex library section
- **row_ids** (*list[int]*, *optional*) – List of row IDS to delete

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_all_user_history(user_id, row_ids=None)

Delete all Tautulli history for a specific user

Parameters

- **user_id** (*str*) – ID of the Plex user
- **row_ids** (*list[int]*, *optional*) – List of row IDs to delete

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_cache()

Delete the cache directory

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_export(export_id, delete_all=False)

Delete exports from Tautulli

Parameters

- **export_id** (*int*) – Row ID of the exported file to delete

- **delete_all** (*bool*) – Whether to delete all exported files (default: False)

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_history(*row_ids*)

Delete history rows from Tautulli

Parameters

row_ids (*list[int]*) – List of row IDs to delete

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_hosted_images(*rating_key=None, service=None, delete_all=False*)

Delete images uploaded to image hosting services

Parameters

- **rating_key** (*int, optional*) – Rating key of image
- **service** (*str, optional*) – Service to delete image from (i.e. 'imgur', 'cloudinary')
- **delete_all** (*bool, optional*) – Whether to delete all images from the service (default: False)

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_image_cache()

Delete image cache directory

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_library(*server_id, section_id, row_ids=None*)

Delete library section from Tautulli.

Also erases library history.

Parameters

- **server_id** (*str*) – Plex server identifier of the library section
- **section_id** (*str*) – ID of the Plex library section
- **row_ids** (*list[int], optional*) – List of row IDs to delete

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_login_log()

Delete the Tautulli login logs

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_lookup_info(*rating_key=None, service=None, delete_all=False*)

Delete the 3rd party API lookup info

Parameters

- **rating_key** (*int, optional*) – rating key of image to delete
- **service** (*str, optional*) – service to delete from (i.e. ‘themoviedb’, ‘tvmaze’, ‘musicbrainz’)
- **delete_all** (*bool, optional*) – Whether to delete all images from the service (default: *False*)

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_media_info_cache(*section_id*)

Delete media info table cache for a specific library

Parameters

section_id (*str*) – ID of the Plex library section

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_mobile_device(*mobile_device_id=None, device_id=None*)

Remove a mobile device from the database

Parameters

- **mobile_device_id** (*int, optional*) – Mobile device database ID to delete
- **device_id** (*str, optional*) – Unique device identifier for the mobile device

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_newsletter(*newsletter_id*)

Remove a newsletter from the database

Parameters

newsletter_id (*int*) – ID of the newsletter to delete

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_newsletter_log()

Delete the Tautulli newsletter logs

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

delete_notification_log()

Delete the Tautulli notification logs

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

delete_notifier(notifier_id)

Remove a notifier from the database

Parameters**notifier_id** (*int*) – ID of the notifier to delete**Returns***True* if successful, *False* if unsuccessful**Return type**

bool

delete_recently_added()

Flush all the recently added items in the database

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

delete_synced_item(client_id, sync_id)

Delete a synced item from a device

Parameters

- **client_id** (*str*) – Client ID of the device to delete from
- **sync_id** (*str*) – Sync ID of the synced item

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

delete_temp_sessions()

Flush all temporary sessions in the database

Returns*True* if successful, *False* if unsuccessful

Return type

bool

delete_user(*user_id*, *row_ids=None*)

Delete a user from Tautulli. Also erases all history of the user.

Parameters

- **user_id** (*str*) – ID of the Plex user
- **row_ids** (*list[int]*, *optional*) – List of row IDs to delete

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

download_config()

Download the Tautulli configuration file

Returns

Config file string

Return type

str

download_database()

Download the Tautulli database file

Returns

Database file byte array

Return type

bytes

download_export(*export_id*)

Download an exported metadata file

Returns

Metadata file byte array

Return type

bytes

download_log(*logfile=None*)

Download the Tautulli log file

Parameters**logfile** (*str*, *optional*) – Log file to download**Returns**

Log file byte array

Return type

bytes

download_plex_log(*logfile=None*)

Download the Plex log file

Parameters**logfile** (*str*, *optional*) – Log file to download

Returns

Log file byte array

Return type

bytes

edit_library(*section_id*, *custom_thumb=None*, *custom_art=None*, *keep_history=True*)

Update a library section on Tautulli

Parameters

- **section_id** (*str*) – ID of the Plex library section
- **custom_thumb** (*str*, *optional*) – URL of the custom library thumbnail
- **custom_art** (*str*, *optional*) – URL of the custom library background art
- **keep_history** (*bool*, *optional*) – Whether to keep library history (default: True)

Returns

True if successful, *False* if unsuccessful

Return type

bool

edit_user(*user_id*, *friendly_name=None*, *custom_thumb=None*, *keep_history=True*, *allow_guest=False*)

Update a user on Tautulli

Parameters

- **user_id** (*str*) – ID of the Plex user
- **friendly_name** (*str*, *optional*) – Friendly name of the user
- **custom_thumb** (*str*, *optional*) – URL of the custom user thumbnail
- **keep_history** (*bool*, *optional*) – Whether to keep user history (default: True)
- **allow_guest** (*bool*, *optional*) – Whether to allow user as a guest (default: False)

Returns

True if successful, *False* if unsuccessful

Return type

bool

export_metadata(*section_id=None*, *user_id=None*, *rating_key=None*, *file_format='csv'*, *metadata_level=1*, *media_info_level=1*, *thumb_level=0*, *art_level=0*, *custom_fields=None*, *export_type=None*, *individual_files=False*)

Export library or media metadata to a file

Parameters

- **section_id** (*int*, *optional*) – Section ID of the library items to export
- **user_id** (*int*, *optional*) – User ID of the playlist items to export
- **rating_key** (*int*, *optional*) – Rating key of the media items to export
- **file_format** (*str*, *optional*) – File format for export (i.e. 'csv', 'json', 'xml', 'm3u') (default: 'csv')
- **metadata_level** (*int*, *optional*) – Level of metadata to export (default: 1)
- **media_info_level** (*int*, *optional*) – Level of media info to export (default: 1)
- **thumb_level** (*int*, *optional*) – Level of poster/cover images to export (default: 0)

- **art_level** (*int*, *optional*) – Level of background artwork images to export (default: 0)
- **custom_fields** (*list[str]*, *optional*) – List of custom fields to export
- **export_type** (*str*, *optional*) – Type of export (i.e. ‘collection’ or ‘playlist’ for library/user export)
- **individual_files** (*bool*, *optional*) – Export each item as an individual field for library/user export (default: False)

Returns

True if successful, *False* if unsuccessful

Return type

bool

get_api_key(*username=None, password=None*)

Get the Tautulli API key. Username and password are required if auth is enabled. Makes and saves the API key if it does not exist.

Parameters

- **username** (*str*, *optional*) – Tautulli username
- **password** (*str*, *optional*) – Tautulli password

Returns

API key

Return type

str or None

get_children_metadata(*rating_key, media_type*)

Get the metadata for the children of a media item.

Parameters

- **rating_key** (*str*) – The rating key of the media item
- **media_type** (*str*) – The type of media item (movie, show, season, episode)

Returns

Dict of data

Return type

dict

get_collections_table(*section_id*)

Get the data on the Tautulli collections tables

Parameters

section_id (*str*) – ID of the Plex library section

Returns

Dict of data

Return type

dict

get_export_fields(*media_type, sub_media_type=None*)

Get a list of available custom export fields

Parameters

- **media_type** (*str*, *optional*) – Media type of the fields to return
- **sub_media_type** (*str*, *optional*) – Child media type for collections (i.e. ‘movie’, ‘show’, ‘video’, ‘audio’, ‘photo’)

Returns

Dict of data

Return type

dict

get_exports_table(*section_id=None, user_id=None, rating_key=None, order_column=None, order_direction=None, start=0, length=25, search=None*)

Get the data on the Tautulli export tables

Parameters

- **section_id** (*str*, *optional*) – ID of the Plex library section
- **user_id** (*str*, *optional*) – ID of the Plex user
- **rating_key** (*str*, *optional*) – Rating key of the exported item
- **order_column** (*str*, *optional*) – Column to order data by (i.e. ‘added_at’, ‘sort_title’, ‘last_played’)
- **order_direction** (*str*, *optional*) – Direction to order the rows (‘desc’ or ‘asc’)
- **start** (*int*, *optional*) – Row number to start from (default: 0)
- **length** (*int*, *optional*) – Number of items to return (default: 25)
- **search** (*str*, *optional*) – String to search for

Returns

Dict of data

Return type

dict

get_geoip_lookup(*ip_address*)

Get the Geolocation info for an IP address

Parameters

ip_address (*str*) – IP address to look up

Returns

Dict of data

Return type

dict

get_history(*grouping=False, include_activity=False, user=None, user_id=None, rating_key=None, parent_rating_key=None, grandparent_rating_key=None, start_date=None, before=None, after=None, section_id=None, media_type=None, transcode_decision=None, guid=None, order_column=None, order_direction=None, start=0, length=25, search=None*)

Get the Tautulli history

Parameters

- **grouping** (*bool*, *optional*) – Whether to group results (default: False)
- **include_activity** (*bool*, *optional*) – Whether to include activity (default: False)
- **user** (*str*, *optional*) – Name of user

- **user_id** (*int, optional*) – ID of user
- **rating_key** (*int, optional*) – Rating key of item
- **parent_rating_key** (*int, optional*) – Parent rating key of item
- **grandparent_rating_key** (*int, optional*) – Grandparent rating key of item
- **start_date** (*datetime, optional*) – Exact date for results
- **before** (*datetime, optional*) – Results before and including the date
- **after** (*datetime, optional*) – Results after and including the date
- **section_id** (*int, optional*) – ID of section
- **media_type** (*str, optional*) – Media type (i.e. ‘movie’, ‘episode’, ‘track’, ‘live’, ‘collection’, ‘playlist’)
- **transcode_decision** (*str, optional*) – Transcode decision (i.e. ‘direct play’, ‘copy’, ‘transcode’)
- **guid** (*str, optional*) – Plex GUID for an item (e.g. “com.plexapp.agents.thetvdb://121361/6/1”)
- **order_column** (*str, optional*) – Column to order data by (i.e. ‘date’, ‘platform’, ‘full_title’)
- **order_direction** (*str, optional*) – Direction to order the rows (‘desc’ or ‘asc’)
- **start** (*int, optional*) – Row number to start from (default: 0)
- **length** (*int, optional*) – Number of items to return (default: 25)
- **search** (*str, optional*) – String to search for

Returns

Dict of data

Return type

dict

get_home_stats(*grouping=False, time_range=30, stats_type='plays', start=0, count=5, stat_id=None, user_id=None, section_id=None*)

Get the homepage watch statistics

Parameters

- **grouping** (*bool, optional*) – Whether to group results (default: False)
- **time_range** (*int, optional*) – Time range to calculate statistics (i.e. 30)
- **stats_type** (*str, optional*) – Type of stats to get (‘plays’ or ‘duration’)
- **start** (*int, optional*) – Row number to start from (default: 0)
- **count** (*int, optional*) – Number of items to return (default: 5)
- **stat_id** (*str, optional*) – Name of a single statistic to return (i.e. ‘top_movies’, ‘popular_tv’, ‘most_concurrent’)
- **user_id** (*int, optional*) – The ID of the Plex user
- **section_id** (*int, optional*) – The ID of the Plex library section

Returns

List of data

Return type

List[dict]

get_item_user_stats(*rating_key*, *grouping=False*, *media_type=None*)

Get the user statistics for the media item

Parameters

- **rating_key** (*str*) – Rating key of the media item
- **grouping** (*bool*, *optional*) – Whether to group results (default: False)
- **media_type** (*str*, *optional*) – Media type of the item (only required for a collection)

Returns

List of data

Return type

List[dict]

get_item_watch_time_stats(*rating_key*, *grouping=False*, *query_days=None*, *media_type=None*)

Get the watch time stats for the media item

Parameters

- **rating_key** (*str*) – Rating key of the media item
- **grouping** (*bool*, *optional*) – Whether to group results (default: False)
- **query_days** (*list[int]*, *optional*) – List of days to get results for (i.e. [0, 1, 14, 30])
- **media_type** (*str*, *optional*) – Media type of the item (only required for a collection)

Returns

Dict of data

Return type

dict

get_libraries_table(*grouping=False*, *order_column=None*, *order_direction=None*, *start=0*, *length=25*, *search=None*)

Get the data on the Tautulli libraries table

Parameters

- **grouping** (*bool*, *optional*) – Whether to group results (default: False)
- **order_column** (*str*, *optional*) – Column to order rows by (i.e. 'section_name', 'count', 'last_played')
- **order_direction** (*str*, *optional*) – Direction to order rows by ('desc' or 'asc')
- **start** (*int*, *optional*) – Row number to start from (default: 0)
- **length** (*int*, *optional*) – Number of items to return (default: 25)
- **search** (*str*, *optional*) – String to search for

Returns

Dict of data

Return type

dict

get_library(*section_id*, *include_last_accessed=False*)

Get a library's details

Parameters

- **section_id** (*str*) – ID of the Plex library section
- **include_last_accessed** (*bool*, *optional*) – Whether to include the last_accessed value for the library (default: False)

Returns

Dict of data

Return type

dict

get_library_media_info(*section_id=None*, *rating_key=None*, *section_type=None*, *order_column=None*, *order_direction=None*, *start=0*, *length=25*, *search=None*, *refresh=False*)

Get the data on the Tautulli media info tables.

Parameters

- **section_id** (*str*) – ID of the Plex library section
- **rating_key** (*str*) – Grandparent or parent rating key
- **section_type** (*str*) – Type of section (i.e. 'movie', 'show', 'artist', 'photo')
- **order_column** (*str*, *optional*) – Column to order rows by (i.e. 'added_at', 'sort_title', 'file_size')
- **order_direction** (*str*, *optional*) – Direction to order rows ('desc' or 'asc')
- **start** (*int*, *optional*) – Row number to start from (default: 0)
- **length** (*int*, *optional*) – Number of items to return (default: 25)
- **search** (*str*, *optional*) – String to search for
- **refresh** (*bool*, *optional*) – Whether to refresh the media info table (default: False)

Returns

Dict of data

Return type

dict

get_library_user_stats(*section_id*, *grouping=False*)

Get a library's user statistics

Parameters

- **section_id** (*str*) – ID of the Plex library section
- **grouping** (*bool*, *optional*) – Whether to group results (default: False)

Returns

Dict of data

Return type

dict

get_library_watch_time_stats(*section_id*, *grouping=False*, *query_days=None*)

Get a library's watch time statistics

Parameters

- **section_id** (*str*) – ID of the Plex library section
- **grouping** (*bool*, *optional*) – Whether to group results (default: False)
- **query_days** (*list[int]*, *optional*) – List of days to get results for (i.e. [0, 1, 14, 30])

Returns

Dict of data

Return type

dict

get_logs(*sort=None, search=None, order_direction=None, regex=None, start=None, end=None*)

Get the Tautulli logs

Parameters

- **sort** (*str*, *optional*) – What to sort the logs by (i.e. 'time', 'thread', 'msg', 'loglevel')
- **search** (*str*, *optional*) – String to search for
- **order_direction** (*str*, *optional*) – Direction to order rows ('desc' or 'asc')
- **regex** (*str*, *optional*) – Regex string to search for
- **start** (*int*, *optional*) – Row number to start from
- **end** (*int*, *optional*) – Row number to end at

Returns

List of data

Return type

List[dict]

get_metadata(*rating_key=None, sync_id=None*)

Get the metadata for a media item

Parameters

- **rating_key** (*str*, *optional*) – Rating key of the media item
- **sync_id** (*str*, *optional*) – Sync ID of a synced item

Returns

Dict of data

Return type

dict

get_new_rating_keys(*rating_key, media_type*)

Get a list of new rating keys for the Plex Media Server of all the item's parent/children

Parameters

- **rating_key** (*str*) – Rating key of item
- **media_type** (*str*) – Type of media (i.e. 'movie', 'show', 'episode', 'album', 'track')

Returns

Dict of data

Return type

dict

get_newsletter_config(*newsletter_id*)

Get the configuration for an existing newsletter agent

Parameters

newsletter_id (*int*) – ID of the newsletter

Returns

Dict of data

Return type

dict

get_newsletter_log(*order_column=None, order_direction=None, start=0, length=25, search=None*)

Get the data on the Tautulli newsletter logs table

Parameters

- **order_column** (*str, optional*) – Column to order rows by (i.e. ‘timestamp’, ‘newsletter_id’, ‘start_date’)
- **order_direction** (*str, optional*) – Direction to order rows (‘desc’ or ‘asc’)
- **start** (*int, optional*) – Row number to start from (default: 0)
- **length** (*int, optional*) – Number of items to return (default: 25)
- **search** (*str, optional*) – String to search for

Returns

Dict of data

Return type

dict

get_notification_log(*order_column=None, order_direction=None, start=0, length=25, search=None*)

Get the data on the Tautulli notification logs table

Parameters

- **order_column** (*str, optional*) – Column to order rows by (i.e. ‘timestamp’, ‘agent_name’, ‘notifier_id’)
- **order_direction** (*str, optional*) – Direction to order rows (‘desc’ or ‘asc’)
- **start** (*int, optional*) – Row number to start from (default: 0)
- **length** (*int, optional*) – Number of items to return (default: 25)
- **search** (*str, optional*) – String to search for

Returns

Dict of data

Return type

dict

get_notifier_config(*notifier_id*)

Get the configuration for an existing notification agent

Parameters

notifier_id (*int*) – ID of the notifier

Returns

Dict of data

Return type

dict

get_notifiers(*notify_action=None*)

Get a list of configured notifiers

Parameters**notify_action** (*str*, *optional*) – The notification action to filter out**Returns**

List of data

Return type

List[dict]

get_old_rating_keys(*rating_key*, *media_type*)

Get a list of old rating keys from the Tautulli database for all the item's parent/children

Parameters

- **rating_key** (*str*) – Rating key of item
- **media_type** (*str*) – Type of media (i.e. 'movie', 'show', 'episode', 'album', 'track')

Returns

Dict of data

Return type

dict

get_playlists_table(*section_id=None*, *user_id=None*)

Get the data on the Tautulli playlists tables

Parameters

- **section_id** (*str*, *optional*) – Section ID of the Plex library
- **user_id** (*str*, *optional*) – User ID of the Plex user

Returns

Dict of data

Return type

dict

get_plays_by_date(*time_range=None*, *y_axis=None*, *user_ids=None*, *grouping=False*)

Get graph data by date

Parameters

- **time_range** (*int*, *optional*) – Number of days of data to return
- **y_axis** (*str*, *optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str]*, *optional*) – List of user IDs to filter data
- **grouping** (*bool*, *optional*) – Whether to group the results (default: False)

Returns

Dict of data

Return type

dict

get_plays_by_day_of_week(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by day of the week

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data
- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

Dict of data

Return type

dict

get_plays_by_hour_of_day(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by hour of the day

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data
- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

Dict of data

Return type

dict

get_plays_by_source_resolution(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by source resolution

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data
- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

Dict of data

Return type

dict

get_plays_by_stream_resolution(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by stream resolution

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data

- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

Dict of data

Return type

dict

get_plays_by_stream_type(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by stream type

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data
- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

Dict of data

Return type

dict

get_plays_by_top_10_platforms(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by top 10 platforms

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data
- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

Dict of data

Return type

dict

get_plays_by_top_10_users(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by top 10 users

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data
- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

Dict of data

Return type

dict

get_plays_per_month(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by month

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data
- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

Dict of data

Return type

dict

get_plex_log(*window=None, logfile=None*)

Get the Plex Media Server logs

Parameters

- **window** (*int, optional*) – Number of tail lines to return
- **logfile** (*str, optional*) – Log file to download

Returns

Dict of data

Return type

dict

get_recently_added(*count, start=0, media_type=None, section_id=None*)

Get all items that were recently added to Plex

Parameters

- **count** (*int*) – Number of item to return
- **start** (*int, optional*) – Item number to start from
- **media_type** (*str, optional*) – Media type (i.e. 'movie', 'show', 'artist')
- **section_id** (*str, optional*) – ID of the Plex library section

Returns

Dict of data

Return type

dict

get_server_id(*hostname, port, ssl=False, remote=False*)

Get the Plex Media Server identifier

Parameters

- **hostname** (*str*) – IP address of the Plex Media Server
- **port** (*int*) – Port of the Plex Media Server
- **ssl** (*bool, optional*) – Whether to use SSL (default: False)
- **remote** (*bool, optional*) – Whether the Plex Media Server is remote (default: False)

Returns

Dict of data

Return type

dict

get_server_pref(*pref*)

Get a specified Plex Media Server preference

Parameters**pref** (*str*) – Name of preference**Returns**

Value of preference

Return type

str

get_settings(*key=None*)

Get all settings from the config file

Parameters**key** (*str*, *optional*) – Name of a config section to return**Returns**

Dict of data

Return type

dict

get_stream_data(*row_id=None*, *session_key=None*)

Get the details of a stream from history or current stream

Parameters

- **row_id** (*int*, *optional*) – Row ID number for a history item
- **session_key** (*int*, *optional*) – Session key for the current stream

Returns

Dict of data

Return type

dict

get_stream_type_by_top_10_platforms(*time_range=None*, *y_axis=None*, *user_ids=None*, *grouping=False*)

Get graph data by stream type by the top 10 platforms

Parameters

- **time_range** (*int*, *optional*) – Number of days of data to return
- **y_axis** (*str*, *optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str]*, *optional*) – List of user IDs to filter data
- **grouping** (*bool*, *optional*) – Whether to group the results (default: False)

Returns

Dict of data

Return type

dict

get_stream_type_by_top_10_users(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by stream type by the top 10 users

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data
- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

Dict of data

Return type

dict

get_synced_items(*machine_id=None, user_id=None*)

Get a list of synced items on the Plex Media Server

Parameters

- **machine_id** (*str, optional*) – Plex Media Server identifier
- **user_id** (*str, optional*) – ID of the Plex user

Returns

Dict of data

Return type

dict

get_user(*user_id, include_last_seen=False*)

Get a user's details

Parameters

- **user_id** (*str*) – ID of the Plex user
- **include_last_seen** (*bool, optional*) – Whether to include the last_seen value for the user (default: False)

Returns

Dict of data

Return type

dict

get_user_ips(*user_id, order_column=None, order_direction=None, start=0, length=25, search=None*)

Get the data on Tautulli's users IP table

Parameters

- **user_id** (*str*) – ID of the Plex user
- **order_column** (*str, optional*) – Column to order rows by (i.e. 'last_seen', 'ip_address', 'player')
- **order_direction** (*str, optional*) – Direction to order rows ('desc' or 'asc')
- **start** (*int, optional*) – Row number to start from (default: 0)
- **length** (*int, optional*) – Number of items to return (default: 25)

- **search** (*str*, *optional*) – String to search for (e.g. “xxx.xxx.xxx.xxx”)

Returns

Dict of data

Return type

dict

get_user_logins(*user_id*, *order_column=None*, *order_direction=None*, *start=0*, *length=25*, *search=None*)

Get the data on Tautulli’s user login table

Parameters

- **user_id** (*str*) – ID of the Plex user
- **order_column** (*str*, *optional*) – Column to order rows by (i.e. ‘date’, ‘time’, ‘ip_address’)
- **order_direction** (*str*, *optional*) – Direction to order rows (‘desc’ or ‘asc’)
- **start** (*int*, *optional*) – Row number to start from (default: 0)
- **length** (*int*, *optional*) – Number of items to return (default: 25)
- **search** (*str*, *optional*) – String to search for (e.g. “xxx.xxx.xxx.xxx”)

Returns

Dict of data

Return type

dict

get_user_player_stats(*user_id*, *grouping=False*)

Get a user’s player statistics

Parameters

- **user_id** (*str*) – ID of the Plex user
- **grouping** (*bool*, *optional*) – Whether to group results (default: False)

Returns

Dict of data

Return type

dict

get_user_watch_time_stats(*user_id*, *grouping=False*, *query_days=None*)

Get a user’s watch time statistics

Parameters

- **user_id** (*str*) – ID of the Plex user
- **grouping** (*bool*, *optional*) – Whether to group results (default: False)
- **query_days** (*list[int]*, *optional*) – List of days to get results for (e.g. [0, 1, 14, 30])

Returns

Dict of data

Return type

dict

get_users_table(*grouping=False, order_column=None, order_direction=None, start=0, length=25, search=None*)

Get the data on Tautulli's users table

Parameters

- **grouping** (*bool, optional*) – Whether to group results (default: False)
- **order_column** (*str, optional*) – Column to order rows by ('friendly_name', 'ip_address', 'player')
- **order_direction** (*str, optional*) – Direction to order rows ('desc' or 'asc')
- **start** (*int, optional*) – Row number to start from (default: 0)
- **length** (*int, optional*) – Number of items to return (default: 25)
- **search** (*str, optional*) – String to search for

Returns

Dict of data

Return type

dict

get_whois_lookup(*ip_address*)

Get the connection info for an IP address

Parameters

ip_address (*str*) – IP address

Returns

Dict of data

Return type

dict

import_config(*config_file_path, backup=True*)

Import a Tautulli config file

Parameters

- **config_file_path** (*str*) – Full path to the config file to import
- **backup** (*bool, optional*) – Whether to back up the current config before importing (default: True)

Returns

True if successful, *False* if unsuccessful

Return type

bool

import_database(*app, database_file_path, method=None, table_name=None, backup=True, import_ignore_interval=None*)

Import a Tautulli, PlexWatch or Plexivity database into Tautulli

Parameters

- **app** (*str*) – Type of app the database is from ('tautulli', 'plexwatch' or 'plexivity')
- **database_file_path** (*str*) – Full path to the database file to import
- **method** (*str, optional*) – Only if app is 'tautulli', method to import database ('merge' or 'overwrite')

- **table_name** (*str*, *optional*) – Only if app is ‘plexwatch’ or ‘plexivity’, table name to import (‘processed’ or ‘grouped’)
- **backup** (*bool*, *optional*) – Whether to back up the current database before importing (default: True)
- **import_ignore_interval** (*int*, *optional*) – Only if app is ‘plexwatch’ or ‘plexivity’, the minimum number of seconds for a stream to import

Returns

True if successful, *False* if unsuccessful

Return type

bool

logout_user_session(*row_ids*)

Logout Tautulli user sessions

Parameters

row_ids (*list[int]*, *optional*) – List of row IDS to sign out

Returns

True if successful, *False* if unsuccessful

Return type

bool

notify(*notifier_id*, *subject*, *body*, *headers=None*, *script_args=None*)

Send a notification using Tautulli

Parameters

- **notifier_id** (*int*) – ID of the notification agent
- **subject** (*str*) – Subject of the message
- **body** (*str*) – Body of the message
- **headers** (*str*, *optional*) – JSON headers for webhook notifications
- **script_args** (*str*, *optional*) – Arguments for script notifications

Returns

Dict of data

Return type

dict

notify_newsletter(*newsletter_id*, *subject=None*, *body=None*, *message=None*)

Send a newsletter using Tautulli

Parameters

- **newsletter_id** (*int*) – ID of the newsletter agent
- **subject** (*str*, *optional*) – Subject of the newsletter
- **body** (*str*, *optional*) – Body of the newsletter
- **message** (*str*, *optional*) – Message of the newsletter

Returns

Dict of data

Return type

dict

notify_recently_added(*rating_key*, *notifier_id=None*)

Send a recently added notification using Tautulli

Parameters

- **rating_key** (*int*) – Rating key for the media item
- **notifier_id** (*int*, *optional*) – ID of the notification agent. Notification will be sent to all enabled notification agents if *notifier_id* is not provided.

Returns

True if successful, *False* if unsuccessful

Return type

bool

pms_image_proxy(*img=None*, *rating_key=None*, *width=None*, *height=None*, *opacity=None*, *background_hex=None*, *blur=None*, *img_format=None*, *fallback=None*, *refresh=False*, *return_hash=False*)

Gets an image from the Plex Media Server and saves it to the image cache directory

Parameters

- **img** (*str*, *optional*) – Path of image (e.g. “/library/metadata/153037/thumb/1462175060”)
- **rating_key** (*str*, *optional*) – Rating key of image
- **width** (*int*, *optional*) – Width of image
- **height** (*int*, *optional*) – Height of image
- **opacity** (*int*, *optional*) – Opacity of image
- **background_hex** (*str*, *optional*) – Background hex color
- **blur** (*int*, *optional*) – Blur level of image
- **img_format** (*str*, *optional*) – Format of image (i.e. ‘png’)
- **fallback** (*str*, *optional*) – Fallback of image (i.e. ‘poster’, ‘cover’, ‘art’, ‘poster-live’, ‘art-live’, ‘art-live-full’)
- **refresh** (*bool*, *optional*) – Whether or refresh the image cache (default: False)
- **return_hash** (*bool*, *optional*) – Whether to return the self-hosted image hash instead of the image (default: False)

Returns

True if successful, *False* if unsuccessful

Return type

bool

refresh_libraries_list()

Refresh the Tautulli libraries list

Returns

True if successful, *False* if unsuccessful

Return type

bool

refresh_users_list()

Refresh the Tautulli users list

Returns

True if successful, *False* if unsuccessful

Return type

bool

register_device(*device_id*, *device_name*, *platform=None*, *version=None*, *friendly_name=None*, *onesignal_id=None*, *min_version=None*)

Register the Tautulli Android App for notifications

Parameters

- **device_id** (*str*) – Unique device identifier for the mobile device
- **device_name** (*str*) – Device name of the mobile device
- **platform** (*str*, *optional*) – The platform of the mobile devices
- **version** (*str*, *optional*) – The version of the app
- **friendly_name** (*str*, *optional*) – Friendly name to identity the mobile device
- **onesignal_id** (*str*, *optional*) – The OneSignal ID of the mobile device
- **min_version** (*str*, *optional*) – The minimum Tautulli version supported by the mobile device (e.g. “v2.5.6”)

Returns

Dict of data

Return type

dict

regroup_history()

Regroup play history in the database

Returns

True if successful, *False* if unsuccessful

Return type

bool

restart()

Restart Tautulli

Returns

True if successful, *False* if unsuccessful

Return type

bool

search(*query*, *limit=None*)

Get search results from the Plex Media Server

Parameters

- **query** (*str*) – String to search for
- **limit** (*int*, *optional*) – Maximum number of items to return per media type

Returns

Dict of data

Return type

dict

set_mobile_device_config(*mobile_device_id*, *friendly_name*=None)

Configure an existing notification agent

Parameters

- **mobile_device_id** (*int*) – ID of the mobile device config to update
- **friendly_name** (*str*, *optional*) – Friendly name to identify the mobile device

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

set_newsletter_config(*newsletter_id*, *agent_id*, ***kwargs*)

Configure an existing newsletter agent

Parameters

- **newsletter_id** (*int*) – ID of the newsletter config to update
- **agent_id** (*int*) – Type of the newsletter to update

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

set_notifier_config(*agent*, *notifier_id*, *agent_id*, ***kwargs*)

Configure an existing notification agent

Parameters

- **agent** (*str*) – Type of the notification agent to update
- **notifier_id** (*int*) – ID of the notifier config to update
- **agent_id** (*int*) – Agent of the notifier to update

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

sql(*query*)

Query the Tautulli database with raw SQL.

Automatically makes a backup of the database if the latest backup is older than 24 hours.

api_sql must be manually enabled in the config file while Tautulli is shut down.**Parameters****query** (*str*) – SQL query**Returns**

Dict of data

Return type

dict

status(*check=None*)

Get the current status of Tautulli

Parameters

check (*str*, *optional*) – What to check (i.e. ‘database’)

Returns

Dict of data

Return type

dict

terminate_session(*session_key=None*, *session_id=None*, *message=None*)

Stop a streaming session

Parameters

- **session_key** (*int*, *optional*) – Session key of the session to terminate
- **session_id** (*str*, *optional*) – Session ID of the session to terminate
- **message** (*str*, *optional*) – Custom message to send to the client

Returns

True if successful, *False* if unsuccessful

Return type

bool

undelete_library(*section_id*, *section_name*)

Restore a deleted library section to Tautulli

Parameters

- **section_id** (*str*) – ID of the Plex library section
- **section_name** (*str*, *optional*) – Name of the Plex library section

Returns

True if successful, *False* if unsuccessful

Return type

bool

undelete_user(*user_id*, *username*)

Restore a deleted user to Tautulli

Parameters

- **user_id** (*str*) – ID of the Plex user
- **username** (*str*) – Username of the Plex user

Returns

True if successful, *False* if unsuccessful

Return type

bool

update()

Update Tautulli

Returns

True if successful, *False* if unsuccessful

Return type

bool

update_metadata_details(*old_rating_key*, *new_rating_key*, *media_type*)

Update the metadata in the Tautulli database by matching rating keys.

Also updates all parents or children of the media item if it is a show/season/episode or artist/album/track.

Parameters

- **old_rating_key** (*str*) – Old rating key for item
- **new_rating_key** (*str*) – New rating key for item
- **media_type** (*str*) – Type of media (i.e. 'movie', 'show', 'episode', 'album', 'track')

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

property arnold: **str**

Get to the chopper!

Returns

Random Arnold Schwarzenegger quote

Return type

str

property date_formats: **dict**

Get the data and time formats used by Tautulli

Returns

Dict of data

Return type

dict

property docs: **dict**

Get the Tautulli API docs as a dict where commands are keys, docstring are value

Returns

Dict of data

Return type

dict

property docs_md: **str**

Get the Tautulli API docs formatted with markdown

Returns

API docs str

Return type

str

property libraries: **List[dict]**

Get a list of all libraries on your server

Returns

List of data

Return type

List[dict]

property library_names: List[dict]

Get list of library names and IDs on the Plex Media Server

Returns

List of names

Return type

list[str]

property newsletters: List[dict]

Get a list of configured newsletters

Returns

List of data

Return type

List[dict]

property notifier_parameters: List[dict]

Get a list of available notification parameters

Returns

List of data

Return type

List[dict]

property pms_update: dict

Check for updates to the Plex Media Server

Returns

Dict of data

Return type

dict

property server_friendly_name: str

Get the name of the Plex Media Server

Returns

Name of the Plex Media Server

Return type

str

property server_identity: dict

Get info about the local server

Returns

Dict of data

Return type

dict

property server_info: dict

Get the Plex Media Server information

Returns

Dict of data

Return type
dict

property server_list: List[dict]

Get all your servers that are published to Plex.tv

Returns
List of data

Return type
List[dict]

property server_status: dict

Get the current status of Tautulli's connection to the Plex server

Returns
Dict of data

Return type
dict

property servers_info: List[dict]

Get info about the Plex Media Server

Returns
List of data

Return type
List[dict]

property shortcuts: *APIShortcuts*

Shortcuts for common API actions

Returns
Access to API shortcuts

Return type
APIShortcuts

property tautulli_info: dict

Get the Tautulli server information

Returns
Dict of data

Return type
dict

property update_check: dict

Check for Tautulli updates

Returns
Dict of data

Return type
dict

property user_names: List[dict]

Get a list of all usernames and user ids

Returns
List of data

Return type

List[dict]

property users: List[dict]

Get a list of all users that have access to your server

Returns

List of data

Return type

List[dict]

2.2 The ObjectAPI class

class tautulli.api.object_api.**ObjectAPI**(*base_url, api_key, verbose=False, verify=True, ssl_verify=True*)

Bases: object

activity(*session_key=None, session_id=None*)

Get the current activity on the Plex Media Server

Parameters

- **session_key** (*int, optional*) – Session key for the session info to return
- **session_id** (*str, optional*) – Session ID of the session info to return

Return type*Activity***Returns**

Activity object

add_newsletter_config(*agent_id*)

Add a new newsletter notification agent

Parameters**agent_id** (*int*) – Newsletter type to add**Returns***True* if successful, *False* if unsuccessful**Return type**

bool

add_notifier_config(*agent_id*)

Add a new notifier notification agent

Parameters**agent_id** (*int*) – Notification agent to add**Type**

int

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

backup_config()

Backup the config.ini file

Returns

True if successful, *False* if unsuccessful

Return type

bool

backup_database()

Backup the plexpy.db database

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_all_library_history(server_id, section_id, row_ids=None)

Delete all Tautulli history for a specific library

Parameters

- **server_id** (*str*) – Plex server identifier of the library section
- **section_id** (*str*) – ID of the Plex library section
- **row_ids** (*list[int]*, *optional*) – List of row IDS to delete

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_all_user_history(user_id, row_ids=None)

Delete all Tautulli history for a specific user

Parameters

- **user_id** (*str*) – ID of the Plex user
- **row_ids** (*list[int]*, *optional*) – List of row IDs to delete

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_cache()

Delete the cache directory

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_export(export_id, delete_all=False)

Delete exports from Tautulli

Parameters

- **export_id** (*int*) – Row ID of the exported file to delete

- **delete_all** (*bool*) – Whether to delete all exported files (default: *False*)

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_history(*row_ids*)

Delete history rows from Tautulli

Parameters

row_ids (*list[int]*) – List of row IDs to delete

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_hosted_images(*rating_key=None, service=None, delete_all=False*)

Delete images uploaded to image hosting services

Parameters

- **rating_key** (*int, optional*) – Rating key of image
- **service** (*str, optional*) – Service to delete image from (i.e. 'imgur', 'cloudinary')
- **delete_all** (*bool, optional*) – Whether to delete all images from the service (default: *False*)

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_image_cache()

Delete image cache directory

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_library(*server_id, section_id, row_ids=None*)

Delete library section from Tautulli.

Also erases library history.

Parameters

- **server_id** (*str*) – Plex server identifier of the library section
- **section_id** (*str*) – ID of the Plex library section
- **row_ids** (*list[int], optional*) – List of row IDs to delete

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_login_log()

Delete the Tautulli login logs

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_lookup_info(*rating_key=None, service=None, delete_all=False*)

Delete the 3rd party API lookup info

Parameters

- **rating_key** (*int, optional*) – rating key of image to delete
- **service** (*str, optional*) – service to delete from (i.e. ‘themoviedb’, ‘tvmaze’, ‘musicbrainz’)
- **delete_all** (*bool, optional*) – Whether to delete all images from the service (default: *False*)

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_media_info_cache(*section_id*)

Delete media info table cache for a specific library

Parameters

section_id (*str*) – ID of the Plex library section

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_mobile_device(*mobile_device_id=None, device_id=None*)

Remove a mobile device from the database

Parameters

- **mobile_device_id** (*int, optional*) – Mobile device database ID to delete
- **device_id** (*str, optional*) – Unique device identifier for the mobile device

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_newsletter(*newsletter_id*)

Remove a newsletter from the database

Parameters

newsletter_id (*int*) – ID of the newsletter to delete

Returns

True if successful, *False* if unsuccessful

Return type

bool

delete_newsletter_log()

Delete the Tautulli newsletter logs

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

delete_notification_log()

Delete the Tautulli notification logs

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

delete_notifier(notifier_id)

Remove a notifier from the database

Parameters**notifier_id** (*int*) – ID of the notifier to delete**Returns***True* if successful, *False* if unsuccessful**Return type**

bool

delete_recently_added()

Flush all the recently added items in the database

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

delete_synced_item(client_id, sync_id)

Delete a synced item from a device

Parameters

- **client_id** (*str*) – Client ID of the device to delete from
- **sync_id** (*str*) – Sync ID of the synced item

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

delete_temp_sessions()

Flush all temporary sessions in the database

Returns*True* if successful, *False* if unsuccessful

Return type

bool

delete_user(*user_id*, *row_ids=None*)

Delete a user from Tautulli. Also erases all history of the user.

Parameters

- **user_id** (*str*) – ID of the Plex user
- **row_ids** (*list[int]*, *optional*) – List of row IDs to delete

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

download_config()

Download the Tautulli configuration file

Returns

Config file string

Return type

str

download_database()

Download the Tautulli database file

Returns

Database file bytearray

Return type

bytearray

download_export(*export_id*)

Download an exported metadata file

Returns

Metadata file byte array

Return type

bytes

download_log()

Download the Tautulli log file

Returns

Log file bytearray

Return type

bytearray

download_plex_log()

Download the Plex log file

Returns

Log file bytearray

Return type

bytearray

edit_library(*section_id*, *custom_thumb=None*, *custom_art=None*, *keep_history=True*)

Update a library section on Tautulli

Parameters

- **section_id** (*str*) – ID of the Plex library section
- **custom_thumb** (*str*, *optional*) – URL of the custom library thumbnail
- **custom_art** (*str*, *optional*) – URL of the custom library background art
- **keep_history** (*bool*, *optional*) – Whether to keep library history (default: True)

Returns

True if successful, *False* if unsuccessful

Return type

bool

edit_user(*user_id*, *friendly_name=None*, *custom_thumb=None*, *keep_history=True*, *allow_guest=False*)

Update a user on Tautulli

Parameters

- **user_id** (*str*) – ID of the Plex user
- **friendly_name** (*str*, *optional*) – Friendly name of the user
- **custom_thumb** (*str*, *optional*) – URL of the custom user thumbnail
- **keep_history** (*bool*, *optional*) – Whether to keep user history (default: True)
- **allow_guest** (*bool*, *optional*) – Whether to allow user as a guest (default: False)

Returns

True if successful, *False* if unsuccessful

Return type

bool

export_metadata(*section_id=None*, *user_id=None*, *rating_key=None*, *file_format='csv'*, *metadata_level=1*, *media_info_level=1*, *thumb_level=0*, *art_level=0*, *custom_fields=None*, *export_type=None*, *individual_files=False*)

Export library or media metadata to a file

Parameters

- **section_id** (*int*, *optional*) – Section ID of the library items to export
- **user_id** (*int*, *optional*) – User ID of the playlist items to export
- **rating_key** (*int*, *optional*) – Rating key of the media items to export
- **file_format** (*str*, *optional*) – File format for export (i.e. 'csv', 'json', 'xml', 'm3u') (default: 'csv')
- **metadata_level** (*int*, *optional*) – Level of metadata to export (default: 1)
- **media_info_level** (*int*, *optional*) – Level of media info to export (default: 1)
- **thumb_level** (*int*, *optional*) – Level of poster/cover images to export (default: 0)
- **art_level** (*int*, *optional*) – Level of background artwork images to export (default: 0)
- **custom_fields** (*list[str]*, *optional*) – List of custom fields to export

- **export_type** (*str*, *optional*) – Type of export (i.e. ‘collection’ or ‘playlist’ for library/user export)
- **individual_files** (*bool*, *optional*) – Export each item as an individual field for library/user export (default: False)

Returns

True if successful, *False* if unsuccessful

Return type

bool

get_api_key(*username=None, password=None*)

Get the Tautulli API key. Username and password are required if auth is enabled. Makes and saves the API key if it does not exist.

Parameters

- **username** (*str*, *optional*) – Tautulli username
- **password** (*str*, *optional*) – Tautulli password

Returns

API key

Return type

str or None

get_children_metadata(*rating_key, media_type*)

Get the metadata for the children of a media item.

Parameters

- **rating_key** (*str*) – The rating key of the media item
- **media_type** (*str*) – The type of media item (movie, show, season, episode)

Returns

ChildrenMetadata object

Return type

ChildrenMetadata

get_collections_table(*section_id*)

Get the data on the Tautulli collections tables

Parameters

section_id (*str*) – ID of the Plex library section

Return type

CollectionsTable

Returns

CollectionsTable object

get_export_fields(*media_type, sub_media_type=None*)

Get a list of available custom export fields

Parameters

- **media_type** (*str*, *optional*) – Media type of the fields to return
- **sub_media_type** (*str*, *optional*) – Child media type for collections (i.e. ‘movie’, ‘show’, ‘video’, ‘audio’, ‘photo’)

Return type*ExportFields***Returns**

ExportFields

get_exports_table(*section_id=None, user_id=None, rating_key=None, order_column=None, order_direction=None, start=0, length=25, search=None*)

Get the data on the Tautulli export tables

Parameters

- **section_id** (*str, optional*) – ID of the Plex library section
- **user_id** (*str, optional*) – ID of the Plex user
- **rating_key** (*str, optional*) – Rating key of the exported item
- **order_column** (*str, optional*) – Column to order data by (i.e. 'added_at', 'sort_title', 'last_played')
- **order_direction** (*str, optional*) – Direction to order the rows ('desc' or 'asc')
- **start** (*int, optional*) – Row number to start from (default: 0)
- **length** (*int, optional*) – Number of items to return (default: 25)
- **search** (*str, optional*) – String to search for

Returns

ExportsTable object

Return type

ExportsTable

get_geoip_lookup(*ip_address*)

Get the Geolocation info for an IP address

Parameters

ip_address (*str*) – IP address to look up

Return type*GeoIPLookup***Returns**

GeoIPLookup

get_history(*grouping=False, include_activity=False, user=None, user_id=None, rating_key=None, parent_rating_key=None, grandparent_rating_key=None, start_date=None, section_id=None, media_type=None, transcode_decision=None, guid=None, order_column=None, order_direction=None, start=0, length=25, search=None*)

Get the Tautulli history

Parameters

- **grouping** (*bool, optional*) – Whether to group results (default: False)
- **include_activity** (*bool, optional*) – Whether to include activity (default: False)
- **user** (*str, optional*) – Name of user
- **user_id** (*int, optional*) – ID of user
- **rating_key** (*int, optional*) – Rating key of item
- **parent_rating_key** (*int, optional*) – Parent rating key of item

- **grandparent_rating_key** (*int*, *optional*) – Grandparent rating key of item
- **start_date** (*datetime*, *optional*) – Date to start results from
- **section_id** (*int*, *optional*) – ID of section
- **media_type** (*str*, *optional*) – Media type (i.e. ‘movie’, ‘episode’, ‘track’, ‘live’, ‘collection’, ‘playlist’)
- **transcode_decision** (*str*, *optional*) – Transcode decision (i.e. ‘direct play’, ‘copy’, ‘transcode’)
- **guid** (*str*, *optional*) – Plex GUID for an item (e.g. “com.plexapp.agents.thetvdb://121361/6/1”)
- **order_column** (*str*, *optional*) – Column to order data by (i.e. ‘date’, ‘platform’, ‘full_title’)
- **order_direction** (*str*, *optional*) – Direction to order the rows (‘desc’ or ‘asc’)
- **start** (*int*, *optional*) – Row number to start from (default: 0)
- **length** (*int*, *optional*) – Number of items to return (default: 25)
- **search** (*str*, *optional*) – String to search for

Return type*History***Returns**

History object

get_home_stats(*grouping=False*, *time_range=30*, *stats_type='plays'*, *start=0*, *count=5*, *stat_id=None*, *user_id=None*, *section_id=None*)

Get the homepage watch statistics

Parameters

- **grouping** (*bool*, *optional*) – Whether to group results (default: False)
- **time_range** (*int*, *optional*) – Time range to calculate statistics (i.e. 30)
- **stats_type** (*str*, *optional*) – Type of stats to get (‘plays’ or ‘duration’)
- **start** (*int*, *optional*) – Row number to start from (default: 0)
- **count** (*int*, *optional*) – Number of items to return (default: 5)
- **stat_id** (*str*, *optional*) – Name of a single statistic to return (i.e. ‘top_movies’, ‘popular_tv’, ‘most_concurrent’)
- **user_id** (*int*, *optional*) – The ID of the Plex user
- **section_id** (*int*, *optional*) – The ID of the Plex library section

Return type*List[HomeStat]***Returns**

List of HomeStat object

get_item_user_stats(*rating_key*, *grouping=False*, *media_type=None*)

Get the user statistics for the media item

Parameters

- **rating_key** (*str*) – Rating key of the media item

- **grouping** (*bool, optional*) – Whether to group results (default: False)
- **media_type** (*str, optional*) – Media type of the item (only required for a collection)

Returns

List of ItemUserStat objects

Return type

List[ItemUserStat]

get_item_watch_time_stats(*rating_key, grouping=False, query_days=None, media_type=None*)

Get the watch time stats for the media item

Parameters

- **rating_key** (*str*) – Rating key of the media item
- **grouping** (*bool, optional*) – Whether to group results (default: False)
- **query_days** (*list[int], optional*) – List of days to get results for (i.e. [0, 1, 14, 30])
- **media_type** (*str, optional*) – Media type of the item (only required for a collection)

Returns

List of ItemWatchTimeStat objects

Return type

List[ItemWatchTimeStat]

get_libraries_table(*grouping=False, order_column=None, order_direction=None, start=0, length=25, search=None*)

Get the data on the Tautulli libraries table

Parameters

- **grouping** (*bool, optional*) – Whether to group results (default: False)
- **order_column** (*str, optional*) – Column to order rows by (i.e. 'section_name', 'count', 'last_played')
- **order_direction** (*str, optional*) – Direction to order rows by ('desc' or 'asc')
- **start** (*int, optional*) – Row number to start from (default: 0)
- **length** (*int, optional*) – Number of items to return (default: 25)
- **search** (*str, optional*) – String to search for

Return type

[LibrariesTable](#)

Returns

LibrariesTable object

get_library(*section_id*)

Get a library's details

Parameters

section_id (*str*) – ID of the Plex library section

Return type

[Library](#)

Returns

Library object

get_library_media_info(*section_id*, *rating_key*, *section_type=None*, *order_column=None*,
order_direction=None, *start=0*, *length=25*, *search=None*, *refresh=False*)

Get the data on the Tautulli media info tables.

Parameters

- **section_id** (*str*) – ID of the Plex library section
- **rating_key** (*str*) – Grandparent or parent rating key
- **section_type** (*str*) – Type of section (i.e. ‘movie’, ‘show’, ‘artist’, ‘photo’)
- **order_column** (*str*, *optional*) – Column to order rows by (i.e. ‘added_at’, ‘sort_title’, ‘file_size’)
- **order_direction** (*str*, *optional*) – Direction to order rows (‘desc’ or ‘asc’)
- **start** (*int*, *optional*) – Row number to start from (default: 0)
- **length** (*int*, *optional*) – Number of items to return (default: 25)
- **search** (*str*, *optional*) – String to search for
- **refresh** (*bool*, *optional*) – Whether to refresh the media info table (default: False)

Return type

LibraryMediaInfo

Returns

LibraryMediaInfo object

get_library_user_stats(*section_id*, *grouping=False*)

Get a library’s user statistics

Parameters

- **section_id** (*str*) – ID of the Plex library section
- **grouping** (*bool*, *optional*) – Whether to group results (default: False)

Return type

LibraryUserStats

Returns

LibraryUserStats objects

get_library_watch_time_stats(*section_id*, *grouping=False*, *query_days=None*)

Get a library’s watch time statistics

Parameters

- **section_id** (*str*) – ID of the Plex library section
- **grouping** (*bool*, *optional*) – Whether to group results (default: False)
- **query_days** (*list[int]*, *optional*) – List of days to get results for (i.e. [0, 1, 14, 30])

Return type

LibraryWatchTimeStats

Returns

LibraryWatchTimeStats object

get_logs(*sort=None, search=None, order_direction=None, regex=None, start=None, end=None*)

Get the Tautulli logs

Parameters

- **sort** (*str, optional*) – What to sort the logs by (i.e. ‘time’, ‘thread’, ‘msg’, ‘loglevel’)
- **search** (*str, optional*) – String to search for
- **order_direction** (*str, optional*) – Direction to order rows (‘desc’ or ‘asc’)
- **regex** (*str, optional*) – Regex string to search for
- **start** (*int, optional*) – Row number to start from
- **end** (*int, optional*) – Row number to end at

Return type

List[[LogEntry](#)]

Returns

List of LogEntry objects

get_metadata(*rating_key=None, sync_id=None*)

Get the metadata for a media item

Parameters

- **rating_key** (*str, optional*) – Rating key of the media item
- **sync_id** (*str, optional*) – Sync ID of a synced item

Return type

[Metadata](#)

Returns

Metadata object

get_new_rating_keys(*rating_key, media_type*)

Get a list of new rating keys for the Plex Media Server of all the item’s parent/children

Parameters

- **rating_key** (*str*) – Rating key of item
- **media_type** (*str*) – Type of media (i.e. ‘movie’, ‘show’, ‘episode’, ‘album’, ‘track’)

Return type

[NewRatingKeys](#)

Returns

NewRatingKeys object

get_newsletter_config(*newsletter_id*)

Get the configuration for an existing newsletter agent

Parameters

newsletter_id (*int*) – ID of the newsletter

Return type

[NewsletterConfig](#)

Returns

NewsletterConfig object

get_newsletter_log(*order_column=None, order_direction=None, start=0, length=25, search=None*)

Get the data on the Tautulli newsletter logs table

Parameters

- **order_column**(*str, optional*) – Column to order rows by (i.e. ‘timestamp’, ‘newsletter_id’, ‘start_date’)
- **order_direction**(*str, optional*) – Direction to order rows (‘desc’ or ‘asc’)
- **start**(*int, optional*) – Row number to start from (default: 0)
- **length**(*int, optional*) – Number of items to return (default: 25)
- **search**(*str, optional*) – String to search for

Return type

[*NewsletterLog*](#)

Returns

NewsletterLog object

get_notification_log(*order_column=None, order_direction=None, start=0, length=25, search=None*)

Get the data on the Tautulli notification logs table

Parameters

- **order_column**(*str, optional*) – Column to order rows by (i.e. ‘timestamp’, ‘agent_name’, ‘notifier_id’)
- **order_direction**(*str, optional*) – Direction to order rows (‘desc’ or ‘asc’)
- **start**(*int, optional*) – Row number to start from (default: 0)
- **length**(*int, optional*) – Number of items to return (default: 25)
- **search**(*str, optional*) – String to search for

Return type

[*NotificationLog*](#)

Returns

NotificationLog object

get_notifier_config(*notifier_id*)

Get the configuration for an existing notification agent

Parameters

notifier_id(*int*) – ID of the notifier

Return type

[*NotifierConfig*](#)

Returns

NotifierConfig object

get_notifiers(*notify_action=None*)

Get a list of configured notifiers

Parameters

notify_action(*str, optional*) – The notification action to filter out

Return type

List[[*Notifier*](#)]

Returns

List of Notifier objects

get_old_rating_keys(*rating_key*, *media_type*)

Get a list of old rating keys from the Tautulli database for all the item's parent/children

Parameters

- **rating_key** (*str*) – Rating key of item
- **media_type** (*str*) – Type of media (i.e. 'movie', 'show', 'episode', 'album', 'track')

Return type*OldRatingKeys***Returns**

OldRatingKeys object

get_playlists_table(*section_id=None*, *user_id=None*)

Get the data on the Tautulli playlists tables

Parameters

- **section_id** (*str*, *optional*) – Section ID of the Plex library
- **user_id** (*str*, *optional*) – User ID of the Plex user

Return type*PlaylistsTable***Returns**

PlaylistsTable object

get_plays_by_date(*time_range=None*, *y_axis=None*, *user_ids=None*, *grouping=False*)

Get graph data by date

Parameters

- **time_range** (*int*, *optional*) – Number of days of data to return
- **y_axis** (*str*, *optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str]*, *optional*) – List of user IDs to filter data
- **grouping** (*bool*, *optional*) – Whether to group the results (default: False)

Returns

PlaysOrStreamTypesBy object

Return type

PlaysOrStreamTypesBy

get_plays_by_day_of_week(*time_range=None*, *y_axis=None*, *user_ids=None*, *grouping=False*)

Get graph data by day of the week

Parameters

- **time_range** (*int*, *optional*) – Number of days of data to return
- **y_axis** (*str*, *optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str]*, *optional*) – List of user IDs to filter data
- **grouping** (*bool*, *optional*) – Whether to group the results (default: False)

Returns

PlaysOrStreamTypesBy object

Return type

PlaysOrStreamTypesBy

get_plays_by_hour_of_day(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by hour of the day

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data
- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

PlaysOrStreamTypesBy object

Return type

PlaysOrStreamTypesBy

get_plays_by_source_resolution(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by source resolution

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data
- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

PlaysOrStreamTypesBy object

Return type

PlaysOrStreamTypesBy

get_plays_by_stream_resolution(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by stream resolution

Parameters

- **time_range** (*int, optional*) – Number of days of data to return
- **y_axis** (*str, optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str], optional*) – List of user IDs to filter data
- **grouping** (*bool, optional*) – Whether to group the results (default: False)

Returns

PlaysOrStreamTypesBy object

Return type

PlaysOrStreamTypesBy

get_plays_by_stream_type(*time_range=None, y_axis=None, user_ids=None, grouping=False*)

Get graph data by stream type

Parameters

- **time_range** (*int, optional*) – Number of days of data to return

- **y_axis** (*str*, *optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str]*, *optional*) – List of user IDs to filter data
- **grouping** (*bool*, *optional*) – Whether to group the results (default: False)

Returns

PlaysOrStreamTypesBy object

Return type

PlaysOrStreamTypesBy

get_plays_by_top_10_platforms(*time_range=None*, *y_axis=None*, *user_ids=None*, *grouping=False*)

Get graph data by top 10 platforms

Parameters

- **time_range** (*int*, *optional*) – Number of days of data to return
- **y_axis** (*str*, *optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str]*, *optional*) – List of user IDs to filter data
- **grouping** (*bool*, *optional*) – Whether to group the results (default: False)

Returns

PlaysOrStreamTypesBy object

Return type

PlaysOrStreamTypesBy

get_plays_by_top_10_users(*time_range=None*, *y_axis=None*, *user_ids=None*, *grouping=False*)

Get graph data by top 10 users

Parameters

- **time_range** (*int*, *optional*) – Number of days of data to return
- **y_axis** (*str*, *optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str]*, *optional*) – List of user IDs to filter data
- **grouping** (*bool*, *optional*) – Whether to group the results (default: False)

Returns

PlaysOrStreamTypesBy object

Return type

PlaysOrStreamTypesBy

get_plays_per_month(*time_range=None*, *y_axis=None*, *user_ids=None*, *grouping=False*)

Get graph data by month

Parameters

- **time_range** (*int*, *optional*) – Number of days of data to return
- **y_axis** (*str*, *optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str]*, *optional*) – List of user IDs to filter data
- **grouping** (*bool*, *optional*) – Whether to group the results (default: False)

Returns

PlaysOrStreamTypesBy object

Return type

PlaysOrStreamTypesBy

get_plex_log(*window=None, log_type=None*)

Get the Plex Media Server logs

Parameters

- **window** (*int, optional*) – Number of tail lines to return
- **log_type** (*str, optional*) – Log type ('server' or 'scanner')

Return type*PlexLog***Returns**

PlexLog object

get_recently_added(*count, start=0, media_type=None, section_id=None*)

Get all items that were recently added to Plex

Parameters

- **count** (*int*) – Number of item to return
- **start** (*int, optional*) – Item number to start from
- **media_type** (*str, optional*) – Media type (i.e. 'movie', 'show', 'artist')
- **section_id** (*str, optional*) – ID of the Plex library section

Return type*RecentlyAdded***Returns**

RecentlyAdded object

get_server_id(*hostname, port, ssl=False, remote=False*)

Get the Plex Media Server identifier

Parameters

- **hostname** (*str*) – IP address of the Plex Media Server
- **port** (*int*) – Port of the Plex Media Server
- **ssl** (*bool, optional*) – Whether to use SSL (default: False)
- **remote** (*bool, optional*) – Whether the Plex Media Server is remote (default: False)

Return type*ServerID***Returns**

ServerID object

get_server_pref(*pref*)

Get a specified Plex Media Server preference

Parameters**pref** (*str*) – Name of preference**Returns**

Value of preference

Return type

str

get_settings(*key=None*)

Get all settings from the config file

Parameters**key** (*str*, *optional*) – Name of a config section to return**Return type***Settings***Returns**

Settings object

get_stream_data(*row_id=None*, *session_key=None*)

Get the details of a stream from history or current stream

Parameters

- **row_id** (*int*, *optional*) – Row ID number for a history item
- **session_key** (*int*, *optional*) – Session key for the current stream

Return type*StreamData***Returns**

StreamData object

get_stream_type_by_top_10_platforms(*time_range=None*, *y_axis=None*, *user_ids=None*, *grouping=False*)

Get graph data by stream type by the top 10 platforms

Parameters

- **time_range** (*int*, *optional*) – Number of days of data to return
- **y_axis** (*str*, *optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str]*, *optional*) – List of user IDs to filter data
- **grouping** (*bool*, *optional*) – Whether to group the results (default: False)

Returns

PlaysOrStreamTypesBy object

Return type

PlaysOrStreamTypesBy

get_stream_type_by_top_10_users(*time_range=None*, *y_axis=None*, *user_ids=None*, *grouping=False*)

Get graph data by stream type by the top 10 users

Parameters

- **time_range** (*int*, *optional*) – Number of days of data to return
- **y_axis** (*str*, *optional*) – Stat type ('plays' or 'duration')
- **user_ids** (*List[str]*, *optional*) – List of user IDs to filter data
- **grouping** (*bool*, *optional*) – Whether to group the results (default: False)

Returns

PlaysOrStreamTypesBy object

Return type

PlaysOrStreamTypesBy

get_synced_items(*machine_id*, *user_id=None*)

Get a list of synced items on the Plex Media Server

Parameters

- **machine_id** (*str*) – Plex Media Server identifier
- **user_id** (*str*, *optional*) – ID of the Plex user

Return type*SyncedItems***Returns**

SyncedItems object

get_user(*user_id*)

Get a user's details

Parameters**user_id** (*str*) – ID of the Plex user**Return type***User***Returns**

User object

get_user_ips(*user_id*, *order_column=None*, *order_direction=None*, *start=0*, *length=25*, *search=None*)

Get the data on Tautulli's users IP table

Parameters

- **user_id** (*str*) – ID of the Plex user
- **order_column** (*str*, *optional*) – Column to order rows by (i.e. 'last_seen', 'ip_address', 'player')
- **order_direction** (*str*, *optional*) – Direction to order rows ('desc' or 'asc')
- **start** (*int*, *optional*) – Row number to start from (default: 0)
- **length** (*int*, *optional*) – Number of items to return (default: 25)
- **search** (*str*, *optional*) – String to search for (e.g. "xxx.xxx.xxx.xxx")

Return type*UserIPs***Returns**

UserIPs object

get_user_logins(*user_id*, *order_column=None*, *order_direction=None*, *start=0*, *length=25*, *search=None*)

Get the data on Tautulli's user login table

Parameters

- **user_id** (*str*) – ID of the Plex user
- **order_column** (*str*, *optional*) – Column to order rows by (i.e. 'date', 'time', 'ip_address')
- **order_direction** (*str*, *optional*) – Direction to order rows ('desc' or 'asc')

- **start** (*int*, *optional*) – Row number to start from (default: 0)
- **length** (*int*, *optional*) – Number of items to return (default: 25)
- **search** (*str*, *optional*) – String to search for (e.g. “xxx.xxx.xxx.xxx”)

Return type*UserLogins***Returns**

UserLogins object

get_user_player_stats(*user_id*, *grouping=False*)

Get a user’s player statistics

Parameters

- **user_id** (*str*) – ID of the Plex user
- **grouping** (*bool*, *optional*) – Whether to group results (default: False)

Return type*UserPlayerStats***Returns**

UserPlayerStats object

get_user_watch_time_stats(*user_id*, *grouping=False*, *query_days=None*)

Get a user’s watch time statistics

Parameters

- **user_id** (*str*) – ID of the Plex user
- **grouping** (*bool*, *optional*) – Whether to group results (default: False)
- **query_days** (*list[int]*, *optional*) – List of days to get results for (e.g. [0, 1, 14, 30])

Return type*UserWatchTimeStats***Returns**

UserWatchTimeStats object

get_users_table(*grouping=False*, *order_column=None*, *order_direction=None*, *start=0*, *length=25*, *search=None*)

Get the data on Tautulli’s users table

Parameters

- **grouping** (*bool*, *optional*) – Whether to group results (default: False)
- **order_column** (*str*, *optional*) – Column to order rows by (‘friendly_name’, ‘ip_address’, ‘player’)
- **order_direction** (*str*, *optional*) – Direction to order rows (‘desc’ or ‘asc’)
- **start** (*int*, *optional*) – Row number to start from (default: 0)
- **length** (*int*, *optional*) – Number of items to return (default: 25)
- **search** (*str*, *optional*) – String to search for

Return type*UsersTable*

Returns

UsersTable object

get_whois_lookup(*ip_address*)

Get the connection info for an IP address

Parameters

ip_address (*str*) – IP address

Return type

WHOISLookup

Returns

WHOISLookup object

import_config(*config_file_path*, *backup=True*)

Import a Tautulli config file

Parameters

- **config_file_path** (*str*) – Full path to the config file to import
- **backup** (*bool*, *optional*) – Whether to back up the current config before importing (default: True)

Returns

True if successful, *False* if unsuccessful

Return type

bool

import_database(*app*, *database_file_path*, *method=None*, *table_name=None*, *backup=True*, *import_ignore_interval=None*)

Import a Tautulli, PlexWatch or Plexivity database into Tautulli

Parameters

- **app** (*str*) – Type of app the database is from ('tautulli', 'plexwatch' or 'plexivity')
- **database_file_path** (*str*) – Full path to the database file to import
- **method** (*str*, *optional*) – Only if app is 'tautulli', method to import database ('merge' or 'overwrite')
- **table_name** (*str*, *optional*) – Only if app is 'plexwatch' or 'plexivity', table name to import ('processed' or 'grouped')
- **backup** (*bool*, *optional*) – Whether to back up the current database before importing (default: True)
- **import_ignore_interval** (*int*, *optional*) – Only if app is 'plexwatch' or 'plexivity', the minimum number of seconds for a stream to import

Returns

True if successful, *False* if unsuccessful

Return type

bool

logout_user_session(*row_ids*)

Logout Tautulli user sessions

Parameters

row_ids (*list[int]*, *optional*) – List of row IDS to sign out

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

notify(*notifier_id*, *subject*, *body*, *headers=None*, *script_args=None*)

Send a notification using Tautulli

Parameters

- **notifier_id** (*int*) – ID of the notification agent
- **subject** (*str*) – Subject of the message
- **body** (*str*) – Body of the message
- **headers** (*str*, *optional*) – JSON headers for webhook notifications
- **script_args** (*str*, *optional*) – Arguments for script notifications

Return type

Notification

Returns

Notification object

notify_newsletter(*newsletter_id*, *subject=None*, *body=None*, *message=None*)

Send a newsletter using Tautulli

Parameters

- **newsletter_id** (*int*) – ID of the newsletter agent
- **subject** (*str*, *optional*) – Subject of the newsletter
- **body** (*str*, *optional*) – Body of the newsletter
- **message** (*str*, *optional*) – Message of the newsletter

Return type

NewsletterNotification

Returns

NewsletterNotification object

notify_recently_added(*rating_key*, *notifier_id=None*)

Send a recently added notification using Tautulli

Parameters

- **rating_key** (*int*) – Rating key for the media item
- **notifier_id** (*int*, *optional*) – ID of the notification agent. Notification will be sent to all enabled notification agents if *notifier_id* is not provided.

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

pms_image_proxy(*img=None*, *rating_key=None*, *width=None*, *height=None*, *opacity=None*,
background_hex=None, *blur=None*, *img_format=None*, *fallback=None*, *refresh=False*,
return_hash=False)

Gets an image from the Plex Media Server and saves it to the image cache directory

Parameters

- **img** (*str, optional*) – Path of image (e.g. “/library/metadata/153037/thumb/1462175060”)
- **rating_key** (*str, optional*) – Rating key of image
- **width** (*int, optional*) – Width of image
- **height** (*int, optional*) – Height of image
- **opacity** (*int, optional*) – Opacity of image
- **background_hex** (*str, optional*) – Background hex color
- **blur** (*int, optional*) – Blur level of image
- **img_format** (*str, optional*) – Format of image (i.e. ‘png’)
- **fallback** (*str, optional*) – Fallback of image (i.e. ‘poster’, ‘cover’, ‘art’, ‘poster-live’, ‘art-live’, ‘art-live-full’)
- **refresh** (*bool, optional*) – Whether or refresh the image cache (default: False)
- **return_hash** (*bool, optional*) – Whether to return the self-hosted image hash instead of the image (default: False)

Returns

True if successful, *False* if unsuccessful

Return type

bool

refresh_libraries_list()

Refresh the Tautulli libraries list

Returns

True if successful, *False* if unsuccessful

Return type

bool

refresh_users_list()

Refresh the Tautulli users list

Returns

True if successful, *False* if unsuccessful

Return type

bool

register_device(device_id, device_name, friendly_name=None, onesignal_id=None, min_version=None)

Register the Tautulli Android App for notifications

Parameters

- **device_id** (*str*) – Unique device identifier for the mobile device
- **device_name** (*str*) – Device name of the mobil device
- **friendly_name** (*str, optional*) – Friendly name to identity the mobile device
- **onesignal_id** (*str, optional*) – The OneSignal ID of the mobile device

- **min_version** (*str*, *optional*) – The minimum Tautulli version supported by the mobile device (e.g. “v2.5.6”)

Return type*RegisteredDevice***Returns**

RegisteredDevice object

regroup_history()

Regroup play history in the database

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

restart()

Restart Tautulli

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

search(*query*, *limit=None*)

Get search results from the Plex Media Server

Parameters

- **query** (*str*) – String to search for
- **limit** (*int*, *optional*) – Maximum number of items to return per media type

Return type*SearchResults***Returns**

SearchResults object

set_mobile_device_config(*mobile_device_id*, *friendly_name=None*)

Configure an existing notification agent

Parameters

- **mobile_device_id** (*int*) – ID of the mobile device config to update
- **friendly_name** (*str*, *optional*) – Friendly name to identify the mobile device

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

set_newsletter_config(*newsletter_id*, *agent_id*, ***kwargs*)

Configure an existing newsletter agent

Parameters

- **newsletter_id** (*int*) – ID of the newsletter config to update
- **agent_id** (*int*) – Type of the newsletter to update

Returns

True if successful, *False* if unsuccessful

Return type

bool

set_notifier_config(*agent*, *notifier_id*, *agent_id*, ***kwargs*)

Configure an existing notification agent

Parameters

- **agent** (*str*) – Type of the notification agent to update
- **notifier_id** (*int*) – ID of the notifier config to update
- **agent_id** (*int*) – Agent of the notifier to update

Returns

True if successful, *False* if unsuccessful

Return type

bool

sql(*query*)

Query the Tautulli database with raw SQL.

Automatically makes a backup of the database if the latest backup is older than 24 hours.

api_sql must be manually enabled in the config file while Tautulli is shut down.

Parameters

query (*str*) – SQL query

Return type

None

Returns

SQLResults object

status(*check=None*)

Get the current status of Tautulli

Parameters

check (*str*, *optional*) – What to check (i.e. ‘database’)

Returns

Status object

Return type

Status

terminate_session(*session_key=None*, *session_id=None*, *message=None*)

Stop a streaming session

Parameters

- **session_key** (*int*, *optional*) – Session key of the session to terminate
- **session_id** (*str*, *optional*) – Session ID of the session to terminate
- **message** (*str*, *optional*) – Custom message to send to the client

Returns

True if successful, *False* if unsuccessful

Return type

bool

undeleter_library(*section_id*, *section_name*)

Restore a deleted library section to Tautulli

Parameters

- **section_id** (*str*) – ID of the Plex library section
- **section_name** (*str*, *optional*) – Name of the Plex library section

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

undeleter_user(*user_id*, *username*)

Restore a deleted user to Tautulli

Parameters

- **user_id** (*str*) – ID of the Plex user
- **username** (*str*) – Username of the Plex user

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

update()

Update Tautulli

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

update_metadata_details(*old_rating_key*, *new_rating_key*, *media_type*)

Update the metadata in the Tautulli database by matching rating keys.

Also updates all parents or children of the media item if it is a show/season/episode or artist/album/track.

Parameters

- **old_rating_key** (*str*) – Old rating key for item
- **new_rating_key** (*str*) – New rating key for item
- **media_type** (*str*) – Type of media (i.e. 'movie', 'show', 'episode', 'album', 'track')

Returns*True* if successful, *False* if unsuccessful**Return type**

bool

property arnold: **str**

Get to the chopper!

Returns

Random Arnold Schwarzenegger quote

Return type

str

property date_formats: *DateFormats*

Get the data and time formats used by Tautulli

Returns

DateFormats object

property docs: *Docs*

Get the Tautulli API docs

Returns

Docs object

property docs_md: str

Get the Tautulli API docs formatted with markdown

Returns

API docs str

Return type

str

property libraries: List[*LibrariesEntry*]

Get a list of all libraries on your server

Returns

List of LibrariesEntry objects

property library_names: List[*LibraryName*]

Get list of library names and IDs on the Plex Media Server

Returns

List of LibraryName objects

property newsletters: List[*Newsletter*]

Get a list of configured newsletters

Returns

List of Newsletter objects

property notifier_parameters: List[*NotifierParameter*]

Get a list of available notification parameters

Returns

List of NotifierParameter objects

property pms_update: *PMSUpdate*

Check for updates to the Plex Media Server

Returns

PMSUpdate object

property server_friendly_name: str

Get the name of the Plex Media Server

Returns

Name of the Plex Media Server

Return type

str

property server_identity: *ServerIdentity*

Get info about the local server

Returns

ServerIdentity object

property server_info: *ServerInfo*

Get the Plex Media Server information

Returns

ServerInfo object

property server_list: List[*ServerListEntry*]

Get all your servers that are published to Plex.tv

Returns

List of ServerListEntry objects

property server_status: *ServerStatus*

Get the current status of Tautulli's connection to the Plex server

Returns

ServerStatus object

Return type*ServerStatus***property servers_info:** List[*ServersInfoEntry*]

Get info about the Plex Media Server

Returns

List of ServersInfoEntry objects

property shortcuts: *APIShortcuts*

Shortcuts for common API actions

Returns

Access to API shortcuts

Return type*APIShortcuts***property tautulli_info:** *TautulliInfo*

Get the Tautulli server information

Returns

Dict of data

Return type

dict

property update_check: *UpdateCheck*

Check for Tautulli updates

Returns

UpdateCheck object

property user_names: List[[UserName](#)]

Get a list of all usernames and user ids

Returns

List of [UserName](#) objects

property users: List[[User](#)]

Get a list of all users that have access to your server

Returns

List of [User](#) objects

2.3 Models

2.3.1 Activity

```
class tautulli.models.activity.Activity(**data)
```

```
    lan_bandwidth: Optional[int]
```

```
    model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to [\[ConfigDict\]\[pydantic.config.ConfigDict\]](#).

```
    model_fields: ClassVar[dict[str, FieldInfo]] = {'lan_bandwidth':
FieldInfo(annotation=Union[int, NoneType], required=True), 'sessions':
FieldInfo(annotation=Union[List[tautulli.models.activity.Session], NoneType],
required=True), 'stream_count': FieldInfo(annotation=Union[str, NoneType],
required=True), 'stream_count_direct_play': FieldInfo(annotation=Union[int,
NoneType], required=True), 'stream_count_direct_stream':
FieldInfo(annotation=Union[int, NoneType], required=True), 'stream_count_transcode':
FieldInfo(annotation=Union[int, NoneType], required=True), 'total_bandwidth':
FieldInfo(annotation=Union[int, NoneType], required=True), 'wan_bandwidth':
FieldInfo(annotation=Union[int, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to [\[FieldInfo\]\[pydantic.fields.FieldInfo\]](#).

This replaces `Model.__fields__` from Pydantic V1.

```
    sessions: Optional[List[Session]]
```

```
    stream_count: Optional[str]
```

```
    stream_count_direct_play: Optional[int]
```

```
    stream_count_direct_stream: Optional[int]
```

```
    stream_count_transcode: Optional[int]
```

```
    property summary: ActivitySummary
```

```
    total_bandwidth: Optional[int]
```

```
    wan_bandwidth: Optional[int]
```

```
class tautulli.models.activity.ActivitySummary(**data)
```

lan_bandwidth: Optional[int]

property message

Get activity summary message

Returns

Summary message

Return type

Optional[str]

model_config: ClassVar[ConfigDict] = {}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'lan_bandwidth': FieldInfo(annotation=Union[int, NoneType], required=False, default=0), 'stream_count': FieldInfo(annotation=Union[str, NoneType], required=False, default='0'), 'total_bandwidth': FieldInfo(annotation=Union[int, NoneType], required=False, default=0), 'transcode_count': FieldInfo(annotation=Union[int, NoneType], required=False, default=0)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

stream_count: Optional[str]

total_bandwidth: Optional[int]

transcode_count: Optional[int]

class tautulli.models.activity.Session(**data)

actors: Optional[List]

added_at: Optional[str]

allow_guest: Optional[int]

art: Optional[str]

aspect_ratio: Optional[str]

audience_rating: Optional[str]

audience_rating_image: Optional[str]

audio_bitrate: Optional[str]

audio_bitrate_mode: Optional[str]

audio_channel_layout: Optional[str]

audio_channels: Optional[str]

audio_codec: Optional[str]

audio_decision: Optional[str]

```
audio_language: Optional[str]
audio_language_code: Optional[str]
audio_profile: Optional[str]
audio_sample_rate: Optional[str]
bandwidth: Optional[str]
banner: Optional[str]
bif_thumb: Optional[str]
bitrate: Optional[str]
channel_call_sign: Optional[str]
channel_identifier: Optional[str]
channel_stream: Optional[int]
channel_thumb: Optional[str]
children_count: Optional[int]
collections: Optional[List]
container: Optional[str]
container_decision: Optional[str]
content_rating: Optional[str]
deleted_user: Optional[int]
device: Optional[str]
directors: Optional[List]
do_notify: Optional[int]
duration: Optional[str]
property duration_milliseconds
email: Optional[str]
property eta
file: Optional[str]
file_size: Optional[str]
friendly_name: Optional[str]
full_title: Optional[str]
genres: Optional[List[str]]
grandparent_guid: Optional[str]
```

```

grandparent_rating_key: Optional[str]
grandparent_thumb: Optional[str]
grandparent_title: Optional[str]
guid: Optional[str]
guids: Optional[List]
height: Optional[str]
property human_bandwidth: str
id: Optional[str]
indexes: Optional[int]
ip_address: Optional[str]
ip_address_public: Optional[str]
is_active: Optional[int]
is_admin: Optional[int]
is_allow_sync: Optional[int]
is_home_user: Optional[int]
is_restricted: Optional[int]
keep_history: Optional[int]
labels: Optional[List]
last_viewed_at: Optional[str]
library_name: Optional[str]
live: Optional[int]
live_uuid: Optional[str]
local: Optional[int]
location: Optional[str]
property location_milliseconds
machine_id: Optional[str]
media_index: Optional[str]
media_type: Optional[str]
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'actors':
FieldInfo(annotation=Union[List, NoneType], required=True), 'added_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'allow_guest':
FieldInfo(annotation=Union[str, NoneType], required=True), 'art':
FieldInfo(annotation=Union[str, NoneType], required=True), 'aspect_ratio':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating_image':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_bitrate':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_bitrate_mode':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_channel_layout':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_channels':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_codec':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_decision':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_language':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_language_code':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_profile':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_sample_rate':
FieldInfo(annotation=Union[str, NoneType], required=True), 'bandwidth':
FieldInfo(annotation=Union[str, NoneType], required=True), 'banner':
FieldInfo(annotation=Union[str, NoneType], required=True), 'bif_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'bitrate':
FieldInfo(annotation=Union[str, NoneType], required=True), 'channel_call_sign':
FieldInfo(annotation=Union[str, NoneType], required=True), 'channel_identifier':
FieldInfo(annotation=Union[str, NoneType], required=True), 'channel_stream':
FieldInfo(annotation=Union[str, NoneType], required=True), 'channel_thumb':
FieldInfo(annotation=Union[int, NoneType], required=True), 'children_count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'collections':
FieldInfo(annotation=Union[List, NoneType], required=True), 'container':
FieldInfo(annotation=Union[str, NoneType], required=True), 'container_decision':
FieldInfo(annotation=Union[str, NoneType], required=True), 'content_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'deleted_user':
FieldInfo(annotation=Union[int, NoneType], required=True), 'device':
FieldInfo(annotation=Union[str, NoneType], required=True), 'directors':
FieldInfo(annotation=Union[List, NoneType], required=True), 'do_notify':
FieldInfo(annotation=Union[int, NoneType], required=True), 'duration':
FieldInfo(annotation=Union[str, NoneType], required=True), 'email':
FieldInfo(annotation=Union[str, NoneType], required=True), 'file':
FieldInfo(annotation=Union[str, NoneType], required=True), 'file_size':
FieldInfo(annotation=Union[str, NoneType], required=True), 'friendly_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'full_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'genres':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'grandparent_guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_rating_key':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guids':
FieldInfo(annotation=Union[List, NoneType], required=True), 'height':
FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'indexes':
FieldInfo(annotation=Union[int, NoneType], required=True), 'ip_address':
FieldInfo(annotation=Union[str, NoneType], required=True), 'ip_address_public':
FieldInfo(annotation=Union[str, NoneType], required=True), 'is_active':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_admin':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_allow_sync':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_home_user':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_restricted':
FieldInfo(annotation=Union[int, NoneType], required=True), 'keep_history':
FieldInfo(annotation=Union[int, NoneType], required=True), 'labels':
FieldInfo(annotation=Union[List, NoneType], required=True), 'last_viewed_at':

```

Metadata about the fields defined on the model, mapping of field names to [*Field-Info*][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
optimized_version: Optional[int]
optimized_version_profile: Optional[str]
optimized_version_title: Optional[str]
original_title: Optional[str]
originally_available_at: Optional[str]
parent_guid: Optional[str]
parent_media_index: Optional[str]
parent_rating_key: Optional[str]
parent_thumb: Optional[str]
parent_title: Optional[str]
platform: Optional[str]
platform_name: Optional[str]
platform_version: Optional[str]
player: Optional[str]
product: Optional[str]
product_version: Optional[str]
profile: Optional[str]
property progress_marker
progress_percent: Optional[str]
property progress_percentage
quality_profile: Optional[str]
rating: Optional[str]
rating_image: Optional[str]
rating_key: Optional[str]
relayed: Optional[int]
row_id: Optional[int]
section_id: Optional[str]
secure: Optional[int]
```

```
selected: Optional[int]
session_id: Optional[str]
session_key: Optional[str]
shared_libraries: Optional[List[str]]
sort_title: Optional[str]
state: Optional[str]
property status_icon
    Get icon for a stream state

    Returns
        emoji icon
stream_aspect_ratio: Optional[str]
stream_audio_bitrate: Optional[str]
stream_audio_bitrate_mode: Optional[str]
stream_audio_channel_layout: Optional[str]
stream_audio_channel_layout_: Optional[str]
stream_audio_channels: Optional[str]
stream_audio_codec: Optional[str]
stream_audio_decision: Optional[str]
stream_audio_language: Optional[str]
stream_audio_language_code: Optional[str]
stream_audio_sample_rate: Optional[str]
stream_bitrate: Optional[str]
stream_container: Optional[str]
stream_container_decision: Optional[str]
stream_duration: Optional[str]
stream_subtitle_codec: Optional[str]
stream_subtitle_container: Optional[str]
stream_subtitle_decision: Optional[str]
stream_subtitle_forced: Optional[int]
stream_subtitle_format: Optional[str]
stream_subtitle_language: Optional[str]
stream_subtitle_language_code: Optional[str]
```



```
stream_subtitle_location: Optional[str]
stream_subtitle_transient: Optional[int]
stream_video_bit_depth: Optional[str]
stream_video_bitrate: Optional[str]
stream_video_chroma_subsampling: Optional[str]
stream_video_codec: Optional[str]
stream_video_codec_level: Optional[str]
stream_video_color_primaries: Optional[str]
stream_video_color_range: Optional[str]
stream_video_color_space: Optional[str]
stream_video_color_trc: Optional[str]
stream_video_decision: Optional[str]
stream_video_dynamic_range: Optional[str]
stream_video_framerate: Optional[str]
stream_video_full_resolution: Optional[str]
stream_video_height: Optional[str]
stream_video_language: Optional[str]
stream_video_language_code: Optional[str]
stream_video_ref_frames: Optional[str]
stream_video_resolution: Optional[str]
stream_video_scan_type: Optional[str]
stream_video_width: Optional[str]
studio: Optional[str]
subtitle_codec: Optional[str]
subtitle_container: Optional[str]
subtitle_decision: Optional[str]
subtitle_forced: Optional[int]
subtitle_format: Optional[str]
subtitle_language: Optional[str]
subtitle_language_code: Optional[str]
subtitle_location: Optional[str]
```

```
subtitles: Optional[int]
summary: Optional[str]
synced_version: Optional[int]
synced_version_profile: Optional[str]
tagline: Optional[str]
throttled: Optional[str]
thumb: Optional[str]
title: Optional[str]
transcode_audio_channels: Optional[str]
transcode_audio_codec: Optional[str]
transcode_container: Optional[str]
transcode_decision: Optional[str]
transcode_height: Optional[str]
transcode_hw_decode: Optional[str]
transcode_hw_decode_title: Optional[str]
transcode_hw_decoding: Optional[int]
transcode_hw_encode: Optional[str]
transcode_hw_encode_title: Optional[str]
transcode_hw_encoding: Optional[int]
transcode_hw_full_pipeline: Optional[int]
transcode_hw_requested: Optional[int]
transcode_key: Optional[str]
transcode_max_offset_available: Optional[int]
transcode_min_offset_available: Optional[int]
transcode_progress: Optional[int]
transcode_protocol: Optional[str]
transcode_speed: Optional[str]
transcode_throttled: Optional[int]
transcode_video_codec: Optional[str]
transcode_width: Optional[str]
property transcoding_stub
```

```
type: Optional[str]
property type_icon
updated_at: Optional[str]
user: Optional[str]
user_id: Optional[int]
user_rating: Optional[str]
user_thumb: Optional[str]
username: Optional[str]
video_bit_depth: Optional[str]
video_bitrate: Optional[str]
video_chroma_subsampling: Optional[str]
video_codec: Optional[str]
video_codec_level: Optional[str]
video_color_primaries: Optional[str]
video_color_range: Optional[str]
video_color_space: Optional[str]
video_color_trc: Optional[str]
video_decision: Optional[str]
video_dynamic_range: Optional[str]
video_frame_rate: Optional[str]
video_framerate: Optional[str]
video_full_resolution: Optional[str]
video_height: Optional[str]
video_language: Optional[str]
video_language_code: Optional[str]
video_profile: Optional[str]
video_ref_frames: Optional[str]
video_resolution: Optional[str]
video_scan_type: Optional[str]
video_width: Optional[str]
view_offset: Optional[str]
```

width: Optional[str]

writers: Optional[List]

year: Optional[str]

`tautulli.models.activity.build_summary_from_activity_json(activity_data)`

Create an ActivitySummary using Activity JSON data

Parameters

activity_data (*dict*) – Activity JSON data to use for Overview

Returns

ActivitySummary object

Return type

ActivitySummary

`tautulli.models.activity.build_summary_from_activity_object(activity)`

Create an ActivitySummary using an Activity object

Parameters

activity (*Activity*) – Activity object to use for Overview

Returns

ActivitySummary object

Return type

ActivitySummary

2.3.2 Collections Table

`class tautulli.models.collections_table.CollectionsTable(**data)`

data: Optional[List[Datum]]

draw: Optional[int]

model_config: ClassVar[ConfigDict] = {}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'data': FieldInfo(annotation=Union[List[tautulli.models.collections_table.Datum], NoneType], required=True), 'draw': FieldInfo(annotation=Union[int, NoneType], required=True), 'recordsFiltered': FieldInfo(annotation=Union[int, NoneType], required=True), 'recordsTotal': FieldInfo(annotation=Union[int, NoneType], required=True)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

recordsFiltered: Optional[int]

recordsTotal: Optional[int]

`class tautulli.models.collections_table.Datum(**data)`

```

addedAt: Optional[str]
art: Optional[Any]
childCount: Optional[int]
collectionMode: Optional[int]
collectionSort: Optional[int]
contentRating: Optional[str]
guid: Optional[str]
librarySectionID: Optional[str]
librarySectionTitle: Optional[str]
maxYear: Optional[int]
minYear: Optional[int]
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].
model_fields: ClassVar[dict[str, FieldInfo]] = {'addedAt':
FieldInfo(annotation=Union[str, NoneType], required=True), 'art':
FieldInfo(annotation=Union[Any, NoneType], required=True), 'childCount':
FieldInfo(annotation=Union[int, NoneType], required=True), 'collectionMode':
FieldInfo(annotation=Union[int, NoneType], required=True), 'collectionSort':
FieldInfo(annotation=Union[int, NoneType], required=True), 'contentRating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'librarySectionID':
FieldInfo(annotation=Union[str, NoneType], required=True), 'librarySectionTitle':
FieldInfo(annotation=Union[str, NoneType], required=True), 'maxYear':
FieldInfo(annotation=Union[str, NoneType], required=True), 'minYear':
FieldInfo(annotation=Union[int, NoneType], required=True), 'ratingKey':
FieldInfo(annotation=Union[int, NoneType], required=True), 'subtype':
FieldInfo(annotation=Union[int, NoneType], required=True), 'summary':
FieldInfo(annotation=Union[str, NoneType], required=True), 'thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'titleSort':
FieldInfo(annotation=Union[str, NoneType], required=True), 'type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'updatedAt':
FieldInfo(annotation=Union[str, NoneType], required=True)}
    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].
    This replaces Model.__fields__ from Pydantic V1.
ratingKey: Optional[int]
subtype: Optional[str]
summary: Optional[str]
thumb: Optional[str]

```

```
title: Optional[str]
titleSort: Optional[str]
type: Optional[str]
updatedAt: Optional[str]
```

2.3.3 Date Formats

```
class tautulli.models.date_formats.DateFormats(**data)

    date_format: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'date_format':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'time_format':
        FieldInfo(annotation=Union[str, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    time_format: Optional[str]
```

2.3.4 Docs

```
class tautulli.models.docs.Docs(**data)

    add_newsletter_config: Optional[str]

    add_notifier_config: Optional[str]

    arnold: Optional[str]

    backup_config: Optional[str]

    backup_db: Optional[str]

    delete_all_library_history: Optional[str]

    delete_all_user_history: Optional[str]

    delete_cache: Optional[str]

    delete_export: Optional[str]

    delete_history: Optional[str]

    delete_hosted_images: Optional[str]

    delete_image_cache: Optional[str]
```

```
delete_library: Optional[str]
delete_login_log: Optional[str]
delete_lookup_info: Optional[str]
delete_media_info_cache: Optional[str]
delete_mobile_device: Optional[str]
delete_newsletter: Optional[str]
delete_newsletter_log: Optional[str]
delete_notification_log: Optional[str]
delete_notifier: Optional[str]
delete_recently_added: Optional[str]
delete_synced_item: Optional[str]
delete_temp_sessions: Optional[str]
delete_user: Optional[str]
docs: Optional[str]
docs_md: Optional[str]
download_config: Optional[str]
download_database: Optional[str]
download_export: Optional[str]
download_log: Optional[str]
download_plex_log: Optional[str]
edit_library: Optional[str]
edit_user: Optional[str]
export_metadata: Optional[str]
get_activity: Optional[str]
get_apikey: Optional[str]
get_collections_table: Optional[str]
get_date_formats: Optional[str]
get_export_fields: Optional[str]
get_exports_table: Optional[str]
get_geoip_lookup: Optional[str]
get_history: Optional[str]
```

```
get_home_stats: Optional[str]
get_libraries: Optional[str]
get_libraries_table: Optional[str]
get_library: Optional[str]
get_library_media_info: Optional[str]
get_library_names: Optional[str]
get_library_user_stats: Optional[str]
get_library_watch_time_stats: Optional[str]
get_logs: Optional[str]
get_metadata: Optional[str]
get_new_rating_keys: Optional[str]
get_newsletter_config: Optional[str]
get_newsletter_log: Optional[str]
get_newsletters: Optional[str]
get_notification_log: Optional[str]
get_notifier_config: Optional[str]
get_notifier_parameters: Optional[str]
get_notifiers: Optional[str]
get_old_rating_keys: Optional[str]
get_playlists_table: Optional[str]
get_plays_by_date: Optional[str]
get_plays_by_dayofweek: Optional[str]
get_plays_by_hourofday: Optional[str]
get_plays_by_source_resolution: Optional[str]
get_plays_by_stream_resolution: Optional[str]
get_plays_by_stream_type: Optional[str]
get_plays_by_top_10_platforms: Optional[str]
get_plays_by_top_10_users: Optional[str]
get_plays_per_month: Optional[str]
get_plex_log: Optional[str]
get_pms_update: Optional[str]
```

```

get_recently_added: Optional[str]
get_server_friendly_name: Optional[str]
get_server_id: Optional[str]
get_server_identity: Optional[str]
get_server_info: Optional[str]
get_server_list: Optional[str]
get_server_pref: Optional[str]
get_servers_info: Optional[str]
get_settings: Optional[str]
get_stream_data: Optional[str]
get_stream_type_by_top_10_platforms: Optional[str]
get_stream_type_by_top_10_users: Optional[str]
get_synced_items: Optional[str]
get_user: Optional[str]
get_user_ips: Optional[str]
get_user_logins: Optional[str]
get_user_names: Optional[str]
get_user_player_stats: Optional[str]
get_user_watch_time_stats: Optional[str]
get_users: Optional[str]
get_users_table: Optional[str]
get_whois_lookup: Optional[str]
import_config: Optional[str]
import_database: Optional[str]
logout_user_session: Optional[str]
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'add_newsletter_config':
FieldInfo(annotation=Union[str, NoneType], required=True), 'add_notifier_config':
FieldInfo(annotation=Union[str, NoneType], required=True), 'arnold':
FieldInfo(annotation=Union[str, NoneType], required=True), 'backup_config':
FieldInfo(annotation=Union[str, NoneType], required=True), 'backup_db':
FieldInfo(annotation=Union[str, NoneType], required=True),
'delete_all_library_history': FieldInfo(annotation=Union[str, NoneType],
required=True), 'delete_all_user_history': FieldInfo(annotation=Union[str,
NoneType], required=True), 'delete_cache': FieldInfo(annotation=Union[str,
NoneType], required=True), 'delete_export': FieldInfo(annotation=Union[str,
NoneType], required=True), 'delete_history': FieldInfo(annotation=Union[str,
NoneType], required=True), 'delete_hosted_images': FieldInfo(annotation=Union[str,
NoneType], required=True), 'delete_image_cache': FieldInfo(annotation=Union[str,
NoneType], required=True), 'delete_library': FieldInfo(annotation=Union[str,
NoneType], required=True), 'delete_login_log': FieldInfo(annotation=Union[str,
NoneType], required=True), 'delete_lookup_info': FieldInfo(annotation=Union[str,
NoneType], required=True), 'delete_media_info_cache':
FieldInfo(annotation=Union[str, NoneType], required=True), 'delete_mobile_device':
FieldInfo(annotation=Union[str, NoneType], required=True), 'delete_newsletter':
FieldInfo(annotation=Union[str, NoneType], required=True), 'delete_newsletter_log':
FieldInfo(annotation=Union[str, NoneType], required=True),
'delete_notification_log': FieldInfo(annotation=Union[str, NoneType],
required=True), 'delete_notifier': FieldInfo(annotation=Union[str, NoneType],
required=True), 'delete_recently_added': FieldInfo(annotation=Union[str, NoneType],
required=True), 'delete_synced_item': FieldInfo(annotation=Union[str, NoneType],
required=True), 'delete_temp_sessions': FieldInfo(annotation=Union[str, NoneType],
required=True), 'delete_user': FieldInfo(annotation=Union[str, NoneType],
required=True), 'docs': FieldInfo(annotation=Union[str, NoneType], required=True),
'docs_md': FieldInfo(annotation=Union[str, NoneType], required=True),
'download_config': FieldInfo(annotation=Union[str, NoneType], required=True),
'download_database': FieldInfo(annotation=Union[str, NoneType], required=True),
'download_export': FieldInfo(annotation=Union[str, NoneType], required=True),
'download_log': FieldInfo(annotation=Union[str, NoneType], required=True),
'download_plex_log': FieldInfo(annotation=Union[str, NoneType], required=True),
'edit_library': FieldInfo(annotation=Union[str, NoneType], required=True),
'edit_user': FieldInfo(annotation=Union[str, NoneType], required=True),
'export_metadata': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_activity': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_apikey': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_collections_table': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_date_formats': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_export_fields': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_exports_table': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_geoip_lookup': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_history': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_home_stats': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_libraries': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_libraries_table': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_library': FieldInfo(annotation=Union[str, NoneType], required=True),
'get_library_media_info': FieldInfo(annotation=Union[str, NoneType],
required=True), 'get_library_names': FieldInfo(annotation=Union[str, NoneType],
required=True), 'get_library_user_stats': FieldInfo(annotation=Union[str,
NoneType], required=True), 'get_library_watch_time_stats':
FieldInfo(annotation=Union[str, NoneType], required=True), 'get_logs':
FieldInfo(annotation=Union[str, NoneType], required=True), 'get_metadata':
FieldInfo(annotation=Union[str, NoneType], required=True), 'get_new_rating_keys':
FieldInfo(annotation=Union[str, NoneType], required=True), 'get_newsletter_config':
FieldInfo(annotation=Union[str, NoneType], required=True), 'get_newsletter_log':
FieldInfo(annotation=Union[str, NoneType], required=True), 'get_newsletters':
FieldInfo(annotation=Union[str, NoneType], required=True), 'get_notification_log':

```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```

notify: Optional[str]

notify_newsletter: Optional[str]

notify_recently_added: Optional[str]

pms_image_proxy: Optional[str]

refresh_libraries_list: Optional[str]

refresh_users_list: Optional[str]

register_device: Optional[str]

restart: Optional[str]

search: Optional[str]

server_status: Optional[str]

set_mobile_device_config: Optional[str]

set_newsletter_config: Optional[str]

set_notifier_config: Optional[str]

sql: Optional[str]

status: Optional[str]

terminate_session: Optional[str]

undelele_library: Optional[str]

undelele_user: Optional[str]

update: Optional[str]

update_check: Optional[str]

update_metadata_details: Optional[str]

```

2.3.5 Export Fields

```
class tautulli.models.export_fields.ExportFields(**data)
```

```

media_info_fields: Optional[List[MediaInfoField]]

metadata_fields: Optional[List[MetadataField]]

model_config: ClassVar[ConfigDict] = {}

```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'media_info_fields':  
FieldInfo(annotation=Union[List[tautulli.models.export_fields.MediaInfoField],  
NoneType], required=True), 'metadata_fields':  
FieldInfo(annotation=Union[List[tautulli.models.export_fields.MetadataField],  
NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
class tautulli.models.export_fields.MediaInfoField(**data)
```

```
    field: Optional[str]
```

```
    level: Optional[int]
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'field':  
FieldInfo(annotation=Union[str, NoneType], required=True), 'level':  
FieldInfo(annotation=Union[int, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
class tautulli.models.export_fields.MetadataField(**data)
```

```
    field: Optional[str]
```

```
    level: Optional[int]
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'field':  
FieldInfo(annotation=Union[str, NoneType], required=True), 'level':  
FieldInfo(annotation=Union[int, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

2.3.6 GeoIP Lookup

```
class tautulli.models.geo_ip_lookup.GeoIPLookup(**data)
```

```
    accuracy: Optional[Any]
```

```
    city: Optional[str]
```

```
    code: Optional[str]
```

```
    continent: Optional[str]
```

```

country: Optional[str]

latitude: Optional[Union[float, None]]

longitude: Optional[Union[float, None]]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'accuracy':
FieldInfo(annotation=Union[Any, NoneType], required=True), 'city':
FieldInfo(annotation=Union[str, NoneType], required=True), 'code':
FieldInfo(annotation=Union[str, NoneType], required=True), 'continent':
FieldInfo(annotation=Union[str, NoneType], required=True), 'country':
FieldInfo(annotation=Union[str, NoneType], required=True), 'latitude':
FieldInfo(annotation=Union[float, NoneType], required=True), 'longitude':
FieldInfo(annotation=Union[float, NoneType], required=True), 'postal_code':
FieldInfo(annotation=Union[str, NoneType], required=True), 'region':
FieldInfo(annotation=Union[str, NoneType], required=True), 'timezone':
FieldInfo(annotation=Union[str, NoneType], required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

postal_code: Optional[str]

region: Optional[str]

timezone: Optional[str]

```

2.3.7 History

```

class tautulli.models.history.Datum(**data)

    date: Optional[int]

    duration: Optional[int]

    friendly_name: Optional[str]

    full_title: Optional[str]

    grandparent_rating_key: Optional[Union[int, str]]

    grandparent_title: Optional[str]

    group_count: Optional[int]

    group_ids: Optional[str]

    guid: Optional[str]

    id: Optional[int]

    ip_address: Optional[str]

```

```
live: Optional[int]

machine_id: Optional[str]

media_index: Optional[Union[int, str]]

media_type: Optional[str]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'date':
FieldInfo(annotation=Union[int, NoneType], required=True), 'duration':
FieldInfo(annotation=Union[int, NoneType], required=True), 'friendly_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'full_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_rating_key':
FieldInfo(annotation=Union[int, str, NoneType], required=True), 'grandparent_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'group_count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'group_ids':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'ip_address':
FieldInfo(annotation=Union[str, NoneType], required=True), 'live':
FieldInfo(annotation=Union[int, NoneType], required=True), 'machine_id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'media_index':
FieldInfo(annotation=Union[int, str, NoneType], required=True), 'media_type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'original_title':
FieldInfo(annotation=Union[str, NoneType], required=True),
'originally_available_at': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_media_index': FieldInfo(annotation=Union[int, str,
NoneType], required=True), 'parent_rating_key': FieldInfo(annotation=Union[int,
str, NoneType], required=True), 'parent_title': FieldInfo(annotation=Union[str,
NoneType], required=True), 'paused_counter': FieldInfo(annotation=Union[int,
NoneType], required=True), 'percent_complete': FieldInfo(annotation=Union[int,
NoneType], required=True), 'platform': FieldInfo(annotation=Union[str, NoneType],
required=True), 'play_duration': FieldInfo(annotation=Union[int, NoneType],
required=True), 'player': FieldInfo(annotation=Union[str, NoneType],
required=True), 'product': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_key': FieldInfo(annotation=Union[int, NoneType],
required=True), 'reference_id': FieldInfo(annotation=Union[int, NoneType],
required=True), 'row_id': FieldInfo(annotation=Union[int, NoneType],
required=True), 'session_key': FieldInfo(annotation=Union[Any, NoneType],
required=True), 'started': FieldInfo(annotation=Union[int, NoneType],
required=True), 'state': FieldInfo(annotation=Union[Any, NoneType], required=True),
'stopped': FieldInfo(annotation=Union[int, NoneType], required=True), 'thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'transcode_decision':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user_id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user_thumb':
FieldInfo(annotation=Union[int, NoneType], required=True), 'watched_status':
FieldInfo(annotation=Union[str, NoneType], required=True), 'year':
FieldInfo(annotation=Union[float, NoneType], required=True), 'year':
FieldInfo(annotation=Union[int, str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to [*FieldInfo*][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```

original_title: Optional[str]

originally_available_at: Optional[str]

parent_media_index: Optional[Union[int, str]]

parent_rating_key: Optional[Union[int, str]]

parent_title: Optional[str]

paused_counter: Optional[int]

percent_complete: Optional[int]

platform: Optional[str]

play_duration: Optional[int]

player: Optional[str]

product: Optional[str]

rating_key: Optional[int]

reference_id: Optional[int]

row_id: Optional[int]

session_key: Optional[Any]

started: Optional[int]

state: Optional[Any]

stopped: Optional[int]

thumb: Optional[str]

title: Optional[str]

transcode_decision: Optional[str]

user: Optional[str]

user_id: Optional[int]

user_thumb: Optional[str]

watched_status: Optional[float]

year: Optional[Union[int, str]]

class tautulli.models.history.History(**data)
    data: Optional[List[Datum]]

```

```
draw: Optional[int]

filter_duration: Optional[str]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'data':
FieldInfo(annotation=Union[List[tautulli.models.history.Datum], NoneType],
required=True), 'draw': FieldInfo(annotation=Union[int, NoneType], required=True),
'filter_duration': FieldInfo(annotation=Union[str, NoneType], required=True),
'recordsFiltered': FieldInfo(annotation=Union[int, NoneType], required=True),
'recordsTotal': FieldInfo(annotation=Union[int, NoneType], required=True),
'total_duration': FieldInfo(annotation=Union[str, NoneType], required=True)}
    Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].
    This replaces Model.__fields__ from Pydantic V1.

recordsFiltered: Optional[int]

recordsTotal: Optional[int]

total_duration: Optional[str]
```

2.3.8 Home Stats

```
class tautulli.models.home_stats.HomeStat(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'rows':
FieldInfo(annotation=Union[List[tautulli.models.home_stats.Row], NoneType],
required=True), 'stat_id': FieldInfo(annotation=Union[str, NoneType],
required=True), 'stat_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'stat_type': FieldInfo(annotation=Union[str, NoneType],
required=False)}
        Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].
        This replaces Model.__fields__ from Pydantic V1.

    rows: Optional[List[Row]]

    stat_id: Optional[str]

    stat_title: Optional[str]

    stat_type: Optional[str]

class tautulli.models.home_stats.Row(**data)

    art: Optional[str]
```



```

content_rating: Optional[str]

count: Optional[int]

friendly_name: Optional[str]

grandparent_thumb: Optional[str]

guid: Optional[str]

labels: Optional[List]

last_play: Optional[int]

last_watch: Optional[int]

live: Optional[int]

media_type: Optional[str]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'art':
FieldInfo(annotation=Union[str, NoneType], required=False), 'content_rating':
FieldInfo(annotation=Union[str, NoneType], required=False), 'count':
FieldInfo(annotation=Union[int, NoneType], required=False), 'friendly_name':
FieldInfo(annotation=Union[str, NoneType], required=False), 'grandparent_thumb':
FieldInfo(annotation=Union[str, NoneType], required=False), 'guid':
FieldInfo(annotation=Union[str, NoneType], required=False), 'labels':
FieldInfo(annotation=Union[List, NoneType], required=False), 'last_play':
FieldInfo(annotation=Union[int, NoneType], required=False), 'last_watch':
FieldInfo(annotation=Union[int, NoneType], required=False), 'live':
FieldInfo(annotation=Union[int, NoneType], required=False), 'media_type':
FieldInfo(annotation=Union[str, NoneType], required=False), 'platform':
FieldInfo(annotation=Union[str, NoneType], required=False), 'platform_name':
FieldInfo(annotation=Union[str, NoneType], required=False), 'player':
FieldInfo(annotation=Union[str, NoneType], required=False), 'rating_key':
FieldInfo(annotation=Union[int, str, NoneType], required=False), 'row_id':
FieldInfo(annotation=Union[int, str, NoneType], required=False), 'section_id':
FieldInfo(annotation=Union[int, NoneType], required=False), 'started':
FieldInfo(annotation=Union[str, NoneType], required=False), 'stopped':
FieldInfo(annotation=Union[str, NoneType], required=False), 'thumb':
FieldInfo(annotation=Union[str, NoneType], required=False), 'title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'total_duration':
FieldInfo(annotation=Union[int, NoneType], required=False), 'total_plays':
FieldInfo(annotation=Union[int, NoneType], required=False), 'user':
FieldInfo(annotation=Union[str, NoneType], required=False), 'user_id':
FieldInfo(annotation=Union[int, NoneType], required=False), 'user_thumb':
FieldInfo(annotation=Union[str, NoneType], required=False), 'users_watched':
FieldInfo(annotation=Union[int, str, NoneType], required=False)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

```

```
platform: Optional[str]
platform_name: Optional[str]
player: Optional[str]
rating_key: Optional[Union[int, str]]
row_id: Optional[Union[int, str]]
section_id: Optional[int]
started: Optional[str]
stopped: Optional[str]
thumb: Optional[str]
title: Optional[str]
total_duration: Optional[int]
total_plays: Optional[int]
user: Optional[str]
user_id: Optional[int]
user_thumb: Optional[str]
users_watched: Optional[Union[int, str]]
```

2.3.9 Library

```
class tautulli.models.library.Library(**data)
    child_count: Optional[Any]
    count: Optional[int]
    deleted_section: Optional[int]
    do_notify: Optional[int]
    do_notify_created: Optional[int]
    is_active: Optional[int]
    keep_history: Optional[int]
    library_art: Optional[str]
    library_thumb: Optional[str]
    model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].
```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'child_count':
FieldInfo(annotation=Union[Any, NoneType], required=True), 'count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'deleted_section':
FieldInfo(annotation=Union[int, NoneType], required=True), 'do_notify':
FieldInfo(annotation=Union[int, NoneType], required=True), 'do_notify_created':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_active':
FieldInfo(annotation=Union[int, NoneType], required=True), 'keep_history':
FieldInfo(annotation=Union[int, NoneType], required=True), 'library_art':
FieldInfo(annotation=Union[str, NoneType], required=True), 'library_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'parent_count':
FieldInfo(annotation=Union[Any, NoneType], required=True), 'row_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'section_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'section_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'section_type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'server_id':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```

parent_count: Optional[Any]

row_id: Optional[int]

section_id: Optional[int]

section_name: Optional[str]

section_type: Optional[str]

server_id: Optional[str]

```

2.3.10 Libraries

```
class tautulli.models.libraries.LibrariesEntry(**data)
```

```

agent: Optional[str]

art: Optional[str]

child_count: Optional[str]

count: Optional[str]

is_active: Optional[int]

model_config: ClassVar[ConfigDict] = {}

```

Configuration for the model, should be a dictionary conforming to *[ConfigDict]*[pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'agent':
FieldInfo(annotation=Union[str, NoneType], required=True), 'art':
FieldInfo(annotation=Union[str, NoneType], required=True), 'child_count':
FieldInfo(annotation=Union[str, NoneType], required=False), 'count':
FieldInfo(annotation=Union[str, NoneType], required=True), 'is_active':
FieldInfo(annotation=Union[int, NoneType], required=True), 'parent_count':
FieldInfo(annotation=Union[str, NoneType], required=False), 'section_id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'section_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'section_type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'thumb':
FieldInfo(annotation=Union[str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
parent_count: Optional[str]

section_id: Optional[str]

section_name: Optional[str]

section_type: Optional[str]

thumb: Optional[str]
```

2.3.11 Libraries Table

```
class tautulli.models.libraries_table.Datum(**data)
```

```
child_count: Optional[int]

content_rating: Optional[str]

count: Optional[int]

do_notify: Optional[int]

do_notify_created: Optional[int]

duration: Optional[int]

guid: Optional[str]

history_row_id: Optional[int]

is_active: Optional[int]

keep_history: Optional[int]

labels: Optional[List]

last_accessed: Optional[int]

last_played: Optional[str]

library_art: Optional[str]
```

```

library_thumb: Optional[str]

live: Optional[int]

media_index: Optional[Union[int, str]]

media_type: Optional[str]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'child_count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'content_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'do_notify':
FieldInfo(annotation=Union[int, NoneType], required=True), 'do_notify_created':
FieldInfo(annotation=Union[int, NoneType], required=True), 'duration':
FieldInfo(annotation=Union[int, NoneType], required=True), 'guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'history_row_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_active':
FieldInfo(annotation=Union[int, NoneType], required=True), 'keep_history':
FieldInfo(annotation=Union[int, NoneType], required=True), 'labels':
FieldInfo(annotation=Union[List, NoneType], required=True), 'last_accessed':
FieldInfo(annotation=Union[int, NoneType], required=True), 'last_played':
FieldInfo(annotation=Union[str, NoneType], required=True), 'library_art':
FieldInfo(annotation=Union[str, NoneType], required=True), 'library_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'live':
FieldInfo(annotation=Union[int, NoneType], required=True), 'media_index':
FieldInfo(annotation=Union[int, str, NoneType], required=True), 'media_type':
FieldInfo(annotation=Union[str, NoneType], required=True),
'originally_available_at': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_count': FieldInfo(annotation=Union[int, NoneType],
required=True), 'parent_media_index': FieldInfo(annotation=Union[int, str,
NoneType], required=True), 'parent_title': FieldInfo(annotation=Union[str,
NoneType], required=True), 'plays': FieldInfo(annotation=Union[int, NoneType],
required=True), 'rating_key': FieldInfo(annotation=Union[int, NoneType],
required=True), 'row_id': FieldInfo(annotation=Union[int, NoneType],
required=True), 'section_id': FieldInfo(annotation=Union[int, NoneType],
required=True), 'section_name': FieldInfo(annotation=Union[str, NoneType],
required=True), 'section_type': FieldInfo(annotation=Union[str, NoneType],
required=True), 'server_id': FieldInfo(annotation=Union[str, NoneType],
required=True), 'thumb': FieldInfo(annotation=Union[str, NoneType], required=True),
'year': FieldInfo(annotation=Union[int, str, NoneType], required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

originally_available_at: Optional[str]

parent_count: Optional[int]

parent_media_index: Optional[Union[int, str]]

parent_title: Optional[str]

```

```
plays: Optional[int]
rating_key: Optional[int]
row_id: Optional[int]
section_id: Optional[int]
section_name: Optional[str]
section_type: Optional[str]
server_id: Optional[str]
thumb: Optional[str]
year: Optional[Union[int, str]]

class tautulli.models.libraries_table.LibrariesTable(**data)
    data: Optional[List[Datum]]
    draw: Optional[int]
    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].
    model_fields: ClassVar[dict[str, FieldInfo]] = {'data':
        FieldInfo(annotation=Union[List[tautulli.models.libraries_table.Datum], NoneType],
        required=True), 'draw': FieldInfo(annotation=Union[int, NoneType], required=True),
        'recordsFiltered': FieldInfo(annotation=Union[int, NoneType], required=True),
        'recordsTotal': FieldInfo(annotation=Union[int, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].
        This replaces Model.__fields__ from Pydantic V1.
    recordsFiltered: Optional[int]
    recordsTotal: Optional[int]
```

2.3.12 Library Media Info

```
class tautulli.models.library_media_info.Datum(**data)
    added_at: Optional[str]
    audio_channels: Optional[str]
    audio_codec: Optional[str]
    bitrate: Optional[str]
    container: Optional[str]
    file_size: Optional[str]
```

```

grandparent_rating_key: Optional[str]

last_played: Optional[int]

media_index: Optional[str]

media_type: Optional[str]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'added_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_channels':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_codec':
FieldInfo(annotation=Union[str, NoneType], required=True), 'bitrate':
FieldInfo(annotation=Union[str, NoneType], required=True), 'container':
FieldInfo(annotation=Union[str, NoneType], required=True), 'file_size':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_rating_key':
FieldInfo(annotation=Union[str, NoneType], required=True), 'last_played':
FieldInfo(annotation=Union[int, NoneType], required=True), 'media_index':
FieldInfo(annotation=Union[str, NoneType], required=True), 'media_type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'parent_media_index':
FieldInfo(annotation=Union[str, NoneType], required=True), 'parent_rating_key':
FieldInfo(annotation=Union[str, NoneType], required=True), 'play_count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'rating_key':
FieldInfo(annotation=Union[str, NoneType], required=True), 'section_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'section_type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'sort_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'video_codec':
FieldInfo(annotation=Union[str, NoneType], required=True), 'video_framerate':
FieldInfo(annotation=Union[str, NoneType], required=True), 'video_resolution':
FieldInfo(annotation=Union[str, NoneType], required=True), 'year':
FieldInfo(annotation=Union[str, NoneType], required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

parent_media_index: Optional[str]

parent_rating_key: Optional[str]

play_count: Optional[int]

rating_key: Optional[str]

section_id: Optional[int]

section_type: Optional[str]

sort_title: Optional[str]

thumb: Optional[str]

```

```
title: Optional[str]

video_codec: Optional[str]

video_framerate: Optional[str]

video_resolution: Optional[str]

year: Optional[str]

class tautulli.models.library_media_info.LibraryMediaInfo(**data)

    data: Optional[List[Datum]]

    draw: Optional[int]

    filtered_file_size: Optional[int]

    last_refreshed: Optional[int]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'data':
    FieldInfo(annotation=Union[List[tautulli.models.library_media_info.Datum],
    NoneType], required=True), 'draw': FieldInfo(annotation=Union[int, NoneType],
    required=True), 'filtered_file_size': FieldInfo(annotation=Union[int, NoneType],
    required=True), 'last_refreshed': FieldInfo(annotation=Union[int, NoneType],
    required=True), 'recordsFiltered': FieldInfo(annotation=Union[int, NoneType],
    required=True), 'recordsTotal': FieldInfo(annotation=Union[int, NoneType],
    required=True), 'total_file_size': FieldInfo(annotation=Union[int, NoneType],
    required=True)}

        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    recordsFiltered: Optional[int]

    recordsTotal: Optional[int]

    total_file_size: Optional[int]
```

2.3.13 Library Names

```
class tautulli.models.library_names.LibraryName(**data)

    agent: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].
```



```

model_fields: ClassVar[dict[str, FieldInfo]] = {'agent':
FieldInfo(annotation=Union[str, NoneType], required=True), 'section_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'section_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'section_type':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```

section_id: Optional[int]

section_name: Optional[str]

section_type: Optional[str]

```

2.3.14 Library User Stats

```

class tautulli.models.library_user_stats.LibraryUserStats(**data)

```

```

friendly_name: Optional[str]

```

```

model_config: ClassVar[ConfigDict] = {}

```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'friendly_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'total_plays':
FieldInfo(annotation=Union[int, NoneType], required=True), 'user_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'user_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'username':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```

total_plays: Optional[int]

user_id: Optional[int]

user_thumb: Optional[str]

username: Optional[str]

```

2.3.15 Library Watch Time Stats

```

class tautulli.models.library_watch_time_stats.LibraryWatchTimeStats(**data)

```

```

model_config: ClassVar[ConfigDict] = {}

```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'query_days':  
FieldInfo(annotation=Union[int, NoneType], required=True), 'total_plays':  
FieldInfo(annotation=Union[int, NoneType], required=True), 'total_time':  
FieldInfo(annotation=Union[int, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
query_days: Optional[int]  
  
total_plays: Optional[int]  
  
total_time: Optional[int]
```

2.3.16 Logs

```
class tautulli.models.logs.LogEntry(**data)
```

```
    loglevel: Optional[str]
```

```
    model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'loglevel':  
FieldInfo(annotation=Union[str, NoneType], required=True), 'msg':  
FieldInfo(annotation=Union[str, NoneType], required=True), 'thread':  
FieldInfo(annotation=Union[str, NoneType], required=True), 'time':  
FieldInfo(annotation=Union[str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
msg: Optional[str]  
  
thread: Optional[str]  
  
time: Optional[str]
```

2.3.17 Metadata

```
class tautulli.models.metadata.Marker(**data)
```

```
    end_time_offset: Optional[int]
```

```
    final: Optional[bool]
```

```
    first: Optional[bool]
```

```
    id: Optional[int]
```

```
    model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'end_time_offset':
FieldInfo(annotation=Union[int, NoneType], required=True), 'final':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'first':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'start_time_offset':
FieldInfo(annotation=Union[int, NoneType], required=True), 'type':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
start_time_offset: Optional[int]
```

```
type: Optional[str]
```

```
class tautulli.models.metadata.MediaInfoItem(**data)
```

```
aspect_ratio: Optional[str]
```

```
audio_channel_layout: Optional[str]
```

```
audio_channels: Optional[str]
```

```
audio_codec: Optional[str]
```

```
audio_profile: Optional[str]
```

```
bitrate: Optional[str]
```

```
channel_call_sign: Optional[str]
```

```
channel_identifier: Optional[str]
```

```
channel_thumb: Optional[str]
```

```
container: Optional[str]
```

```
height: Optional[str]
```

```
id: Optional[str]
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict]*[pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'aspect_ratio':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_channel_layout':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_channels':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_codec':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_profile':
FieldInfo(annotation=Union[str, NoneType], required=True), 'bitrate':
FieldInfo(annotation=Union[str, NoneType], required=True), 'channel_call_sign':
FieldInfo(annotation=Union[str, NoneType], required=True), 'channel_identifier':
FieldInfo(annotation=Union[str, NoneType], required=True), 'channel_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'container':
FieldInfo(annotation=Union[str, NoneType], required=True), 'height':
FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'optimized_version':
FieldInfo(annotation=Union[int, NoneType], required=True), 'parts':
FieldInfo(annotation=Union[List[tautulli.models.metadata.Part], NoneType],
required=True), 'video_codec': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_framerate': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_full_resolution': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_profile': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_resolution': FieldInfo(annotation=Union[str, NoneType],
required=True), 'width': FieldInfo(annotation=Union[str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to [*FieldInfo*][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
optimized_version: Optional[int]

parts: Optional[List[Part]]

video_codec: Optional[str]

video_framerate: Optional[str]

video_full_resolution: Optional[str]

video_profile: Optional[str]

video_resolution: Optional[str]

width: Optional[str]
```

```
class tautulli.models.metadata.Metadata(**data)

    actors: Optional[List[str]]

    added_at: Optional[str]

    art: Optional[str]

    audience_rating: Optional[str]

    audience_rating_image: Optional[str]

    banner: Optional[str]

    children_count: Optional[int]
```

```

collections: Optional[List]
content_rating: Optional[str]
directors: Optional[List[str]]
duration: Optional[str]
edition_title: Optional[str]
full_title: Optional[str]
genres: Optional[List[str]]
grandparent_guid: Optional[str]
grandparent_guids: Optional[List[str]]
grandparent_rating_key: Optional[str]
grandparent_thumb: Optional[str]
grandparent_title: Optional[str]
grandparent_year: Optional[str]
guid: Optional[str]
guids: Optional[List[str]]
labels: Optional[List]
last_viewed_at: Optional[str]
library_name: Optional[str]
live: Optional[int]
markers: Optional[Marker]
media_index: Optional[str]
media_info: Optional[List[MediaInfoItem]]
media_type: Optional[str]
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'actors':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'added_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'art':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating_image':
FieldInfo(annotation=Union[str, NoneType], required=True), 'banner':
FieldInfo(annotation=Union[str, NoneType], required=True), 'children_count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'collections':
FieldInfo(annotation=Union[List, NoneType], required=True), 'content_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'directors':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'duration':
FieldInfo(annotation=Union[str, NoneType], required=True), 'edition_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'full_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'genres':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'grandparent_guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_guids':
FieldInfo(annotation=Union[List[str], NoneType], required=True),
'grandparent_rating_key': FieldInfo(annotation=Union[str, NoneType],
required=True), 'grandparent_thumb': FieldInfo(annotation=Union[str, NoneType],
required=True), 'grandparent_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'grandparent_year': FieldInfo(annotation=Union[str, NoneType],
required=True), 'guid': FieldInfo(annotation=Union[str, NoneType], required=True),
'guids': FieldInfo(annotation=Union[List[str], NoneType], required=True), 'labels':
FieldInfo(annotation=Union[List, NoneType], required=True), 'last_viewed_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'library_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'live':
FieldInfo(annotation=Union[int, NoneType], required=True), 'markers':
FieldInfo(annotation=Union[Marker, NoneType], required=True), 'media_index':
FieldInfo(annotation=Union[str, NoneType], required=True), 'media_info':
FieldInfo(annotation=Union[List[tautulli.models.metadata.MediaInfoItem], NoneType],
required=True), 'media_type': FieldInfo(annotation=Union[str, NoneType],
required=True), 'original_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'originally_available_at': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_guid': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_guids': FieldInfo(annotation=Union[List[str],
NoneType], required=True), 'parent_media_index': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_rating_key': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_thumb': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_title': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_year': FieldInfo(annotation=Union[str,
NoneType], required=True), 'rating': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_image': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_key': FieldInfo(annotation=Union[str, NoneType],
required=True), 'section_id': FieldInfo(annotation=Union[str, NoneType],
required=True), 'sort_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'studio': FieldInfo(annotation=Union[str, NoneType],
required=True), 'summary': FieldInfo(annotation=Union[str, NoneType],
required=True), 'tagline': FieldInfo(annotation=Union[str, NoneType],
required=True), 'thumb': FieldInfo(annotation=Union[str, NoneType], required=True),
'title': FieldInfo(annotation=Union[str, NoneType], required=True), 'updated_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'writers':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'year':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to [*FieldInfo*][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
original_title: Optional[str]
originally_available_at: Optional[str]
parent_guid: Optional[str]
parent_guids: Optional[List[str]]
parent_media_index: Optional[str]
parent_rating_key: Optional[str]
parent_thumb: Optional[str]
parent_title: Optional[str]
parent_year: Optional[str]
rating: Optional[str]
rating_image: Optional[str]
rating_key: Optional[str]
section_id: Optional[str]
sort_title: Optional[str]
studio: Optional[str]
summary: Optional[str]
tagline: Optional[str]
thumb: Optional[str]
title: Optional[str]
updated_at: Optional[str]
user_rating: Optional[str]
writers: Optional[List[str]]
year: Optional[str]

class tautulli.models.metadata.Part(**data)
    file: Optional[str]
    file_size: Optional[str]
    id: Optional[str]
    indexes: Optional[int]
```

```
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'file':
FieldInfo(annotation=Union[str, NoneType], required=True), 'file_size':
FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'indexes':
FieldInfo(annotation=Union[int, NoneType], required=True), 'selected':
FieldInfo(annotation=Union[int, NoneType], required=True), 'streams':
FieldInfo(annotation=Union[List[tautulli.models.metadata.Stream], NoneType],
required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

selected: Optional[int]

streams: Optional[List[Stream]]

class tautulli.models.metadata.Stream(**data)

    audio_bitrate: Optional[str]

    audio_bitrate_mode: Optional[str]

    audio_channel_layout: Optional[str]

    audio_channels: Optional[str]

    audio_codec: Optional[str]

    audio_language: Optional[str]

    audio_language_code: Optional[str]

    audio_profile: Optional[str]

    audio_sample_rate: Optional[str]

    id: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].
```



```

model_fields: ClassVar[dict[str, FieldInfo]] = {'audio_bitrate':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_bitrate_mode':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_channel_layout':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_channels':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_codec':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_language':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_language_code':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_profile':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_sample_rate':
FieldInfo(annotation=Union[str, NoneType], required=False), 'id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'selected':
FieldInfo(annotation=Union[int, NoneType], required=True), 'type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'video_bit_depth':
FieldInfo(annotation=Union[str, NoneType], required=False), 'video_bitrate':
FieldInfo(annotation=Union[str, NoneType], required=False),
'video_chroma_subsampling': FieldInfo(annotation=Union[str, NoneType],
required=False), 'video_codec': FieldInfo(annotation=Union[str, NoneType],
required=False), 'video_codec_level': FieldInfo(annotation=Union[str, NoneType],
required=False), 'video_color_primaries': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_color_range': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_color_space': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_color_trc': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_dynamic_range': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_frame_rate': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_height': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_language': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_language_code': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_profile': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_ref_frames': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_scan_type': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_width': FieldInfo(annotation=Union[str,
NoneType], required=False)}

```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```

selected: Optional[int]

type: Optional[str]

video_bit_depth: Optional[str]

video_bitrate: Optional[str]

video_chroma_subsampling: Optional[str]

video_codec: Optional[str]

video_codec_level: Optional[str]

video_color_primaries: Optional[str]

video_color_range: Optional[str]

video_color_space: Optional[str]

```

```
video_color_trc: Optional[str]
video_dynamic_range: Optional[str]
video_frame_rate: Optional[str]
video_height: Optional[str]
video_language: Optional[str]
video_language_code: Optional[str]
video_profile: Optional[str]
video_ref_frames: Optional[str]
video_scan_type: Optional[str]
video_width: Optional[str]
```

2.3.18 New Rating Keys

```
class tautulli.models.new_rating_keys.Field0(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'rating_key':
    FieldInfo(annotation=Union[int, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    rating_key: Optional[int]

class tautulli.models.new_rating_keys.NewRatingKeys(**data)

    field_0: Optional[Field0]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'field_0':
    FieldInfo(annotation=Union[Field0, NoneType], required=True, alias='0',
    alias_priority=2)}
        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.
```

2.3.19 Newsletter Config

```
class tautulli.models.newsletter_config.Config(**data)

    custom_cron: Optional[int]

    filename: Optional[str]

    formatted: Optional[int]

    incl_libraries: Optional[List[str]]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'custom_cron':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'filename':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'formatted':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'incl_libraries':
    FieldInfo(annotation=Union[List[str], NoneType], required=True), 'notifier_id':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'save_only':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'threaded':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'time_frame':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'time_frame_units':
    FieldInfo(annotation=Union[str, NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    notifier_id: Optional[int]

    save_only: Optional[int]

    threaded: Optional[int]

    time_frame: Optional[int]

    time_frame_units: Optional[str]

class tautulli.models.newsletter_config.ConfigOption(**data)

    description: Optional[str]

    input_type: Optional[str]

    label: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'description':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'input_type':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'label':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'name':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'select_options':
    FieldInfo(annotation=Union[SelectOptions, NoneType], required=True), 'value':
    FieldInfo(annotation=Union[List[str], NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

`name: Optional[str]`

`select_options: Optional[SelectOptions]`

`value: Optional[List[str]]`

`class tautulli.models.newsletter_config.EmailConfig(**data)`

`bcc: Optional[List]`

`cc: Optional[List]`

`from_: Optional[str]`

`from_name: Optional[str]`

`html_support: Optional[int]`

`model_config: ClassVar[ConfigDict] = {}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

`model_fields: ClassVar[dict[str, FieldInfo]] = {'bcc': FieldInfo(annotation=Union[List, NoneType], required=True), 'cc': FieldInfo(annotation=Union[List, NoneType], required=True), 'from_': FieldInfo(annotation=Union[str, NoneType], required=True, alias='from', alias_priority=2), 'from_name': FieldInfo(annotation=Union[str, NoneType], required=True), 'html_support': FieldInfo(annotation=Union[int, NoneType], required=True), 'notifier_id': FieldInfo(annotation=Union[int, NoneType], required=True), 'smtp_password': FieldInfo(annotation=Union[str, NoneType], required=True), 'smtp_port': FieldInfo(annotation=Union[int, NoneType], required=True), 'smtp_server': FieldInfo(annotation=Union[str, NoneType], required=True), 'smtp_user': FieldInfo(annotation=Union[str, NoneType], required=True), 'tls': FieldInfo(annotation=Union[int, NoneType], required=True), 'to': FieldInfo(annotation=Union[List[str], NoneType], required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

`notifier_id: Optional[int]`

`smtp_password: Optional[str]`

`smtp_port: Optional[int]`

`smtp_server: Optional[str]`

`smtp_user: Optional[str]`

`tls: Optional[int]`

`to: Optional[List[str]]`

```

class tautulli.models.newsletter_config.EmailConfigOption(**data)

    description: Optional[str]

    input_type: Optional[str]

    label: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'description':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'input_type':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'label':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'name':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'select_options':
    FieldInfo(annotation=Union[List[tautulli.models.newsletter_config.SelectOption],
    NoneType], required=False), 'value': FieldInfo(annotation=Union[int, str,
    List[str], NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    name: Optional[str]

    select_options: Optional[List[SelectOption]]

    value: Optional[Union[Union[int, str], List[str]]]

class tautulli.models.newsletter_config.MovieLibrary(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'text':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'value':
    FieldInfo(annotation=Union[int, NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    text: Optional[str]

    value: Optional[int]

class tautulli.models.newsletter_config.MusicLibrary(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'text':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'value':
    FieldInfo(annotation=Union[int, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

`text: Optional[str]`

`value: Optional[int]`

```
class tautulli.models.newsletter_config.NewsletterConfig(**data)
```

`active: Optional[int]`

`agent_id: Optional[int]`

`agent_label: Optional[str]`

`agent_name: Optional[str]`

`body: Optional[str]`

`config: Optional[Config]`

`config_options: Optional[List[ConfigOption]]`

`cron: Optional[str]`

`email_config: Optional[EmailConfig]`

`email_config_options: Optional[List[EmailConfigOption]]`

`friendly_name: Optional[str]`

`id: Optional[int]`

`id_name: Optional[str]`

`model_config: ClassVar[ConfigDict] = {}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'active':
FieldInfo(annotation=Union[int, NoneType], required=True), 'agent_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'agent_label':
FieldInfo(annotation=Union[str, NoneType], required=True), 'agent_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'body':
FieldInfo(annotation=Union[str, NoneType], required=True), 'config':
FieldInfo(annotation=Union[Config, NoneType], required=True), 'config_options':
FieldInfo(annotation=Union[List[tautulli.models.newsletter_config.ConfigOption],
NoneType], required=True), 'cron': FieldInfo(annotation=Union[str, NoneType],
required=True), 'email_config': FieldInfo(annotation=Union[EmailConfig, NoneType],
required=True), 'email_config_options': FieldInfo(annotation=Union[List[tautulli.
models.newsletter_config.EmailConfigOption], NoneType], required=True),
'friendly_name': FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'id_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'subject':
FieldInfo(annotation=Union[str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

subject: `Optional[str]`

class `tautulli.models.newsletter_config.OtherVideoLibrary(**data)`

model_config: `ClassVar[ConfigDict] = {}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'text': FieldInfo(annotation=Union[str, NoneType], required=True), 'value': FieldInfo(annotation=Union[int, NoneType], required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

text: `Optional[str]`

value: `Optional[int]`

class `tautulli.models.newsletter_config.SelectOption(**data)`

model_config: `ClassVar[ConfigDict] = {}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'text': FieldInfo(annotation=Union[str, NoneType], required=True), 'value': FieldInfo(annotation=Union[str, NoneType], required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

text: `Optional[str]`

value: `Optional[str]`

class `tautulli.models.newsletter_config.SelectOptions(**data)`

Movie_Libraries: `Optional[List[MovieLibrary]]`

Music_Libraries: `Optional[List[MusicLibrary]]`

Other_Video_Libraries: `Optional[List[OtherVideoLibrary]]`

TV_Show_Libraries: `Optional[List[TVShowLibrary]]`

model_config: `ClassVar[ConfigDict] = {}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'Movie_Libraries':
FieldInfo(annotation=Union[List[tautulli.models.newsletter_config.MovieLibrary],
NoneType], required=True, alias='Movie Libraries', alias_priority=2),
'Music_Libraries':
FieldInfo(annotation=Union[List[tautulli.models.newsletter_config.MusicLibrary],
NoneType], required=True, alias='Music Libraries', alias_priority=2),
'Other_Video_Libraries': FieldInfo(annotation=Union[List[tautulli.models.
newsletter_config.OtherVideoLibrary], NoneType], required=True, alias='Other Video
Libraries', alias_priority=2), 'TV_Show_Libraries':
FieldInfo(annotation=Union[List[tautulli.models.newsletter_config.TVShowLibrary],
NoneType], required=True, alias='TV Show Libraries', alias_priority=2)}
```

Metadata about the fields defined on the model, mapping of field names to *[Field-Info]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
class tautulli.models.newsletter_config.TVShowLibrary(**data)
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[Config-Dict]*[pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'text':
FieldInfo(annotation=Union[str, NoneType], required=True), 'value':
FieldInfo(annotation=Union[int, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[Field-Info]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
text: Optional[str]
```

```
value: Optional[int]
```

2.3.20 Newsletter Log

```
class tautulli.models.newsletter_log.Datum(**data)
```

```
agent_id: Optional[int]
```

```
agent_name: Optional[str]
```

```
body_text: Optional[str]
```

```
end_date: Optional[str]
```

```
id: Optional[int]
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[Config-Dict]*[pydantic.config.ConfigDict].


```

model_fields: ClassVar[dict[str, FieldInfo]] = {'agent_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'agent_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'body_text':
FieldInfo(annotation=Union[str, NoneType], required=True), 'end_date':
FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'newsletter_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'notify_action':
FieldInfo(annotation=Union[str, NoneType], required=True), 'start_date':
FieldInfo(annotation=Union[str, NoneType], required=True), 'subject_text':
FieldInfo(annotation=Union[str, NoneType], required=True), 'success':
FieldInfo(annotation=Union[int, NoneType], required=True), 'timestamp':
FieldInfo(annotation=Union[int, NoneType], required=True), 'uuid':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```

newsletter_id: Optional[int]

notify_action: Optional[str]

start_date: Optional[str]

subject_text: Optional[str]

success: Optional[int]

timestamp: Optional[int]

uuid: Optional[str]

```

```
class tautulli.models.newsletter_log.NewsletterLog(**data)
```

```

data: Optional[List[Datum]]

draw: Optional[int]

model_config: ClassVar[ConfigDict] = {}

```

Configuration for the model, should be a dictionary conforming to *[ConfigDict]*[pydantic.config.ConfigDict].

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'data':
FieldInfo(annotation=Union[List[tautulli.models.newsletter_log.Datum], NoneType],
required=True), 'draw': FieldInfo(annotation=Union[int, NoneType], required=True),
'recordsFiltered': FieldInfo(annotation=Union[int, NoneType], required=True),
'recordsTotal': FieldInfo(annotation=Union[int, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```

recordsFiltered: Optional[int]

recordsTotal: Optional[int]

```

2.3.21 Newsletter

```
class tautulli.models.newsletter.Newsletter(**data)

    active: Optional[int]

    agent_id: Optional[int]

    agent_label: Optional[str]

    agent_name: Optional[str]

    cron: Optional[str]

    friendly_name: Optional[str]

    id: Optional[int]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'active':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'agent_id':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'agent_label':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'agent_name':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'cron':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'friendly_name':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
    FieldInfo(annotation=Union[int, NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.
```

2.3.22 Notification Log

```
class tautulli.models.notification_log.Datum(**data)

    agent_id: Optional[int]

    agent_name: Optional[str]

    body_text: Optional[str]

    id: Optional[int]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].
```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'agent_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'agent_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'body_text':
FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'notifier_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'notify_action':
FieldInfo(annotation=Union[str, NoneType], required=True), 'rating_key':
FieldInfo(annotation=Union[int, NoneType], required=True), 'session_key':
FieldInfo(annotation=Union[int, NoneType], required=True), 'subject_text':
FieldInfo(annotation=Union[str, NoneType], required=True), 'success':
FieldInfo(annotation=Union[int, NoneType], required=True), 'timestamp':
FieldInfo(annotation=Union[int, NoneType], required=True), 'user':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user_id':
FieldInfo(annotation=Union[int, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```

notifier_id: Optional[int]
notify_action: Optional[str]
rating_key: Optional[int]
session_key: Optional[int]
subject_text: Optional[str]
success: Optional[int]
timestamp: Optional[int]
user: Optional[str]
user_id: Optional[int]

```

```
class tautulli.models.notification_log.NotificationLog(**data)
```

```

data: Optional[List[Datum]]
draw: Optional[int]

```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'data':
FieldInfo(annotation=Union[List[tautulli.models.notification_log.Datum], NoneType],
required=True), 'draw': FieldInfo(annotation=Union[int, NoneType], required=True),
'recordsFiltered': FieldInfo(annotation=Union[int, NoneType], required=True),
'recordsTotal': FieldInfo(annotation=Union[int, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```

recordsFiltered: Optional[int]
recordsTotal: Optional[int]

```

2.3.23 Notifier Parameters

```
class tautulli.models.notifier_parameters.NotifierParameter(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'name':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'type':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'value':
        FieldInfo(annotation=Union[str, NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    name: Optional[str]

    type: Optional[str]

    value: Optional[str]
```

2.3.24 Notifiers

```
class tautulli.models.notifiers.Notifier(**data)

    active: Optional[int]

    agent_id: Optional[int]

    agent_label: Optional[str]

    agent_name: Optional[str]

    friendly_name: Optional[str]

    id: Optional[int]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'active':
        FieldInfo(annotation=Union[int, NoneType], required=True), 'agent_id':
        FieldInfo(annotation=Union[int, NoneType], required=True), 'agent_label':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'agent_name':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'friendly_name':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
        FieldInfo(annotation=Union[int, NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.
```

2.3.25 Old Rating Keys

```
class tautulli.models.old_rating_keys.Field0(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'rating_key':
    FieldInfo(annotation=Union[int, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    rating_key: Optional[int]

class tautulli.models.old_rating_keys.OldRatingKeys(**data)

    field_0: Optional[Field0]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'field_0':
    FieldInfo(annotation=Union[Field0, NoneType], required=True, alias='0',
    alias_priority=2)}
        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.
```

2.3.26 Playlists Table

```
class tautulli.models.playlists_table.Datum(**data)

    addedAt: Optional[str]

    composite: Optional[str]

    duration: Optional[int]

    guid: Optional[str]

    leafCount: Optional[int]

    librarySectionID: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].
```

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'addedAt':
FieldInfo(annotation=Union[str, NoneType], required=True), 'composite':
FieldInfo(annotation=Union[str, NoneType], required=True), 'duration':
FieldInfo(annotation=Union[int, NoneType], required=True), 'guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'leafCount':
FieldInfo(annotation=Union[int, NoneType], required=True), 'librarySectionID':
FieldInfo(annotation=Union[str, NoneType], required=True), 'playlistType':
FieldInfo(annotation=Union[str, NoneType], required=True), 'ratingKey':
FieldInfo(annotation=Union[int, NoneType], required=True), 'smart':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'summary':
FieldInfo(annotation=Union[str, NoneType], required=True), 'title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'updatedAt':
FieldInfo(annotation=Union[str, NoneType], required=True), 'userID':
FieldInfo(annotation=Union[Any, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
playlistType: Optional[str]
```

```
ratingKey: Optional[int]
```

```
smart: Optional[bool]
```

```
summary: Optional[str]
```

```
title: Optional[str]
```

```
type: Optional[str]
```

```
updatedAt: Optional[str]
```

```
userID: Optional[Any]
```

```
class tautulli.models.playlists_table.PlaylistsTable(**data)
```

```
data: Optional[List[Datum]]
```

```
draw: Optional[int]
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict]*[pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'data':
FieldInfo(annotation=Union[List[tautulli.models.playlists_table.Datum], NoneType],
required=True), 'draw': FieldInfo(annotation=Union[int, NoneType], required=True),
'recordsFiltered': FieldInfo(annotation=Union[int, NoneType], required=True),
'recordsTotal': FieldInfo(annotation=Union[int, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
recordsFiltered: Optional[int]
```

```
recordsTotal: Optional[int]
```

2.3.27 Plex Log

```
class tautulli.models.plex_log.PlexLog(**data)

    data: Optional[List]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'data':
    FieldInfo(annotation=Union[List, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.
```

2.3.28 PMS Update

```
class tautulli.models.pms_update.PMSUpdate(**data)

    changelog_added: Optional[str]

    changelog_fixed: Optional[str]

    distro: Optional[str]

    distro_build: Optional[str]

    download_url: Optional[str]

    extra_info: Optional[str]

    label: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'changelog_added':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'changelog_fixed':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'distro':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'distro_build':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'download_url':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'extra_info':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'label':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'platform':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'release_date':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'requirements':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'update_available':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'version':
    FieldInfo(annotation=Union[str, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.
```

```
platform: Optional[str]
release_date: Optional[int]
requirements: Optional[str]
update_available: Optional[bool]
version: Optional[str]
```

2.3.29 Recently Added

```
class tautulli.models.recently_added.RecentlyAdded(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'recently_added':
        FieldInfo(annotation=Union[List, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    recently_added: Optional[List]
```

2.3.30 Registered Device

```
class tautulli.models.registered_device.RegisteredDevice(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].
```



```

model_fields: ClassVar[dict[str, FieldInfo]] = {'pms_identifier':
FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_ip':
FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_is_cloud':
FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_is_remote':
FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_platform':
FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_plexpass':
FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_port':
FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_ssl':
FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_url':
FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_url_manual':
FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_version':
FieldInfo(annotation=Union[str, NoneType], required=True), 'server_id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'tautulli_branch':
FieldInfo(annotation=Union[str, NoneType], required=True), 'tautulli_commit':
FieldInfo(annotation=Union[str, NoneType], required=True), 'tautulli_install_type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'tautulli_platform':
FieldInfo(annotation=Union[str, NoneType], required=True),
'tautulli_platform_device_name': FieldInfo(annotation=Union[str, NoneType],
required=True), 'tautulli_platform_linux_distro': FieldInfo(annotation=Union[str,
NoneType], required=True), 'tautulli_platform_release':
FieldInfo(annotation=Union[str, NoneType], required=True),
'tautulli_platform_version': FieldInfo(annotation=Union[str, NoneType],
required=True), 'tautulli_python_version': FieldInfo(annotation=Union[str,
NoneType], required=True), 'tautulli_version': FieldInfo(annotation=Union[str,
NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```

pms_identifier: Optional[str]

pms_ip: Optional[str]

pms_is_cloud: Optional[int]

pms_is_remote: Optional[int]

pms_name: Optional[str]

pms_platform: Optional[str]

pms_plexpass: Optional[int]

pms_port: Optional[int]

pms_ssl: Optional[int]

pms_url: Optional[str]

pms_url_manual: Optional[int]

pms_version: Optional[str]

server_id: Optional[str]

```

```
tautulli_branch: Optional[str]
tautulli_commit: Optional[str]
tautulli_install_type: Optional[str]
tautulli_platform: Optional[str]
tautulli_platform_device_name: Optional[str]
tautulli_platform_linux_distro: Optional[str]
tautulli_platform_release: Optional[str]
tautulli_platform_version: Optional[str]
tautulli_python_version: Optional[str]
tautulli_version: Optional[str]
```

2.3.31 Search Results

```
class tautulli.models.search_results.EpisodeItem(**data)
    actors: Optional[List[str]]
    added_at: Optional[str]
    art: Optional[str]
    audience_rating: Optional[str]
    audience_rating_image: Optional[str]
    banner: Optional[str]
    children_count: Optional[int]
    collections: Optional[List]
    content_rating: Optional[str]
    directors: Optional[List[str]]
    duration: Optional[str]
    full_title: Optional[str]
    genres: Optional[List[str]]
    grandparent_guid: Optional[str]
    grandparent_rating_key: Optional[str]
    grandparent_thumb: Optional[str]
    grandparent_title: Optional[str]
    guid: Optional[str]
```

```
guids: Optional[List]
labels: Optional[List]
last_viewed_at: Optional[str]
library_name: Optional[str]
live: Optional[int]
media_index: Optional[str]
media_info: Optional[List[MediaInfoItem]]
media_type: Optional[str]
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].
```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'actors':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'added_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'art':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating_image':
FieldInfo(annotation=Union[str, NoneType], required=True), 'banner':
FieldInfo(annotation=Union[str, NoneType], required=True), 'children_count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'collections':
FieldInfo(annotation=Union[List, NoneType], required=True), 'content_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'directors':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'duration':
FieldInfo(annotation=Union[str, NoneType], required=True), 'full_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'genres':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'grandparent_guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_rating_key':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guids':
FieldInfo(annotation=Union[List, NoneType], required=True), 'labels':
FieldInfo(annotation=Union[List, NoneType], required=True), 'last_viewed_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'library_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'live':
FieldInfo(annotation=Union[int, NoneType], required=True), 'media_index':
FieldInfo(annotation=Union[str, NoneType], required=True), 'media_info':
FieldInfo(annotation=Union[List[tautulli.models.search_results.MediaInfoItem],
NoneType], required=True), 'media_type': FieldInfo(annotation=Union[str, NoneType],
required=True), 'original_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'originally_available_at': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_guid': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_media_index': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_rating_key': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_thumb': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_title': FieldInfo(annotation=Union[str,
NoneType], required=True), 'rating': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_image': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_key': FieldInfo(annotation=Union[str, NoneType],
required=True), 'section_id': FieldInfo(annotation=Union[str, NoneType],
required=True), 'sort_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'studio': FieldInfo(annotation=Union[str, NoneType],
required=True), 'summary': FieldInfo(annotation=Union[str, NoneType],
required=True), 'tagline': FieldInfo(annotation=Union[str, NoneType],
required=True), 'thumb': FieldInfo(annotation=Union[str, NoneType], required=True),
'title': FieldInfo(annotation=Union[str, NoneType], required=True), 'updated_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'writers':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'year':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

original_title: Optional[str]

```
originally_available_at: Optional[str]
parent_guid: Optional[str]
parent_media_index: Optional[str]
parent_rating_key: Optional[str]
parent_thumb: Optional[str]
parent_title: Optional[str]
rating: Optional[str]
rating_image: Optional[str]
rating_key: Optional[str]
section_id: Optional[str]
sort_title: Optional[str]
studio: Optional[str]
summary: Optional[str]
tagline: Optional[str]
thumb: Optional[str]
title: Optional[str]
updated_at: Optional[str]
user_rating: Optional[str]
writers: Optional[List[str]]
year: Optional[str]

class tautulli.models.search_results.MediaInfoItem(**data)
    aspect_ratio: Optional[str]
    audio_channel_layout: Optional[str]
    audio_channels: Optional[str]
    audio_codec: Optional[str]
    audio_profile: Optional[str]
    bitrate: Optional[str]
    channel_call_sign: Optional[str]
    channel_identifier: Optional[str]
    channel_thumb: Optional[str]
    container: Optional[str]
```

```
height: Optional[str]

id: Optional[str]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'aspect_ratio':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_channel_layout':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_channels':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_codec':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_profile':
FieldInfo(annotation=Union[str, NoneType], required=True), 'bitrate':
FieldInfo(annotation=Union[str, NoneType], required=True), 'channel_call_sign':
FieldInfo(annotation=Union[str, NoneType], required=True), 'channel_identifier':
FieldInfo(annotation=Union[str, NoneType], required=True), 'channel_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'container':
FieldInfo(annotation=Union[str, NoneType], required=True), 'height':
FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'optimized_version':
FieldInfo(annotation=Union[int, NoneType], required=True), 'parts':
FieldInfo(annotation=Union[List[tautulli.models.search_results.Part], NoneType],
required=True), 'video_codec': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_framerate': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_full_resolution': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_profile': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_resolution': FieldInfo(annotation=Union[str, NoneType],
required=True), 'width': FieldInfo(annotation=Union[str, NoneType], required=True)}
    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

optimized_version: Optional[int]

parts: Optional[List[Part]]

video_codec: Optional[str]

video_framerate: Optional[str]

video_full_resolution: Optional[str]

video_profile: Optional[str]

video_resolution: Optional[str]

width: Optional[str]

class tautulli.models.search_results.MovieItem(**data)
    actors: Optional[List]
    added_at: Optional[str]
    art: Optional[str]
```

```

audience_rating: Optional[str]
audience_rating_image: Optional[str]
banner: Optional[str]
children_count: Optional[int]
collections: Optional[List]
content_rating: Optional[str]
directors: Optional[List]
duration: Optional[str]
full_title: Optional[str]
genres: Optional[List]
grandparent_guid: Optional[str]
grandparent_rating_key: Optional[str]
grandparent_thumb: Optional[str]
grandparent_title: Optional[str]
guid: Optional[str]
guids: Optional[List]
labels: Optional[List]
last_viewed_at: Optional[str]
library_name: Optional[str]
live: Optional[int]
media_index: Optional[str]
media_info: Optional[List[MediaInfoItem]]
media_type: Optional[str]
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'actors':
FieldInfo(annotation=Union[List, NoneType], required=True), 'added_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'art':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating_image':
FieldInfo(annotation=Union[str, NoneType], required=True), 'banner':
FieldInfo(annotation=Union[str, NoneType], required=True), 'children_count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'collections':
FieldInfo(annotation=Union[List, NoneType], required=True), 'content_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'directors':
FieldInfo(annotation=Union[List, NoneType], required=True), 'duration':
FieldInfo(annotation=Union[str, NoneType], required=True), 'full_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'genres':
FieldInfo(annotation=Union[List, NoneType], required=True), 'grandparent_guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_rating_key':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guids':
FieldInfo(annotation=Union[List, NoneType], required=True), 'labels':
FieldInfo(annotation=Union[List, NoneType], required=True), 'last_viewed_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'library_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'live':
FieldInfo(annotation=Union[int, NoneType], required=True), 'media_index':
FieldInfo(annotation=Union[str, NoneType], required=True), 'media_info':
FieldInfo(annotation=Union[List[tautulli.models.search_results.MediaInfoItem],
NoneType], required=True), 'media_type': FieldInfo(annotation=Union[str, NoneType],
required=True), 'original_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'originally_available_at': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_guid': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_media_index': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_rating_key': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_thumb': FieldInfo(annotation=Union[str,
NoneType], required=True), 'parent_title': FieldInfo(annotation=Union[str,
NoneType], required=True), 'rating': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_image': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_key': FieldInfo(annotation=Union[str, NoneType],
required=True), 'section_id': FieldInfo(annotation=Union[str, NoneType],
required=True), 'sort_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'studio': FieldInfo(annotation=Union[str, NoneType],
required=True), 'summary': FieldInfo(annotation=Union[str, NoneType],
required=True), 'tagline': FieldInfo(annotation=Union[str, NoneType],
required=True), 'thumb': FieldInfo(annotation=Union[str, NoneType], required=True),
'title': FieldInfo(annotation=Union[str, NoneType], required=True), 'updated_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'writers':
FieldInfo(annotation=Union[List, NoneType], required=True), 'year':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

original_title: `Optional[str]`


```

originally_available_at: Optional[str]
parent_guid: Optional[str]
parent_media_index: Optional[str]
parent_rating_key: Optional[str]
parent_thumb: Optional[str]
parent_title: Optional[str]
rating: Optional[str]
rating_image: Optional[str]
rating_key: Optional[str]
section_id: Optional[str]
sort_title: Optional[str]
studio: Optional[str]
summary: Optional[str]
tagline: Optional[str]
thumb: Optional[str]
title: Optional[str]
updated_at: Optional[str]
user_rating: Optional[str]
writers: Optional[List]
year: Optional[str]

```

```
class tautulli.models.search_results.Part(**data)
```

```

file: Optional[str]
file_size: Optional[str]
id: Optional[str]
indexes: Optional[int]
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].
model_fields: ClassVar[dict[str, FieldInfo]] = {'file':
FieldInfo(annotation=Union[str, NoneType], required=True), 'file_size':
FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'indexes':
FieldInfo(annotation=Union[int, NoneType], required=True), 'selected':
FieldInfo(annotation=Union[int, NoneType], required=True), 'streams':
FieldInfo(annotation=Union[List[tautulli.models.search_results.Stream], NoneType],
required=True)}

```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

selected: `Optional[int]`

streams: `Optional[List[Stream]]`

class `tautulli.models.search_results.ResultsList(**data)`

album: `Optional[List]`

artist: `Optional[List]`

collection: `Optional[List]`

episode: `Optional[List[EpisodeItem]]`

model_config: `ClassVar[ConfigDict] = {}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'album': FieldInfo(annotation=Union[List, NoneType], required=True), 'artist': FieldInfo(annotation=Union[List, NoneType], required=True), 'collection': FieldInfo(annotation=Union[List, NoneType], required=True), 'episode': FieldInfo(annotation=Union[List[tautulli.models.search_results.EpisodeItem], NoneType], required=True), 'movie': FieldInfo(annotation=Union[List[tautulli.models.search_results.MovieItem], NoneType], required=True), 'season': FieldInfo(annotation=Union[List[tautulli.models.search_results.SeasonItem], NoneType], required=True), 'show': FieldInfo(annotation=Union[List[tautulli.models.search_results.ShowItem], NoneType], required=True), 'track': FieldInfo(annotation=Union[List, NoneType], required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

movie: `Optional[List[MovieItem]]`

season: `Optional[List[SeasonItem]]`

show: `Optional[List[ShowItem]]`

track: `Optional[List]`

class `tautulli.models.search_results.SearchResults(**data)`

model_config: `ClassVar[ConfigDict] = {}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'results_count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'results_list':
FieldInfo(annotation=Union[ResultsList, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```

results_count: Optional[int]

```

```

results_list: Optional[ResultsList]

```

```

class tautulli.models.search_results.SeasonItem(**data)

```

```

    actors: Optional[List[str]]

```

```

    added_at: Optional[str]

```

```

    art: Optional[str]

```

```

    audience_rating: Optional[str]

```

```

    audience_rating_image: Optional[str]

```

```

    banner: Optional[str]

```

```

    children_count: Optional[int]

```

```

    collections: Optional[List]

```

```

    content_rating: Optional[str]

```

```

    directors: Optional[List]

```

```

    duration: Optional[str]

```

```

    full_title: Optional[str]

```

```

    genres: Optional[List[str]]

```

```

    grandparent_guid: Optional[str]

```

```

    grandparent_rating_key: Optional[str]

```

```

    grandparent_thumb: Optional[str]

```

```

    grandparent_title: Optional[str]

```

```

    guid: Optional[str]

```

```

    guides: Optional[List]

```

```

    labels: Optional[List]

```

```

    last_viewed_at: Optional[str]

```

```

    library_name: Optional[str]

```

```

    live: Optional[int]

```

```

    media_index: Optional[str]

```

```

media_info: Optional[List]

media_type: Optional[str]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'actors':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'added_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'art':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating_image':
FieldInfo(annotation=Union[str, NoneType], required=True), 'banner':
FieldInfo(annotation=Union[str, NoneType], required=True), 'children_count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'collections':
FieldInfo(annotation=Union[List, NoneType], required=True), 'content_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'directors':
FieldInfo(annotation=Union[List, NoneType], required=True), 'duration':
FieldInfo(annotation=Union[str, NoneType], required=True), 'full_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'genres':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'grandparent_guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_rating_key':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guids':
FieldInfo(annotation=Union[List, NoneType], required=True), 'labels':
FieldInfo(annotation=Union[List, NoneType], required=True), 'last_viewed_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'library_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'live':
FieldInfo(annotation=Union[int, NoneType], required=True), 'media_index':
FieldInfo(annotation=Union[str, NoneType], required=True), 'media_info':
FieldInfo(annotation=Union[List, NoneType], required=True), 'media_type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'original_title':
FieldInfo(annotation=Union[str, NoneType], required=True),
'originally_available_at': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_guid': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_media_index': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_rating_key': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_thumb': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_image': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_key': FieldInfo(annotation=Union[str, NoneType],
required=True), 'section_id': FieldInfo(annotation=Union[str, NoneType],
required=True), 'sort_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'studio': FieldInfo(annotation=Union[str, NoneType],
required=True), 'summary': FieldInfo(annotation=Union[str, NoneType],
required=True), 'tagline': FieldInfo(annotation=Union[str, NoneType],
required=True), 'thumb': FieldInfo(annotation=Union[str, NoneType], required=True),
'title': FieldInfo(annotation=Union[str, NoneType], required=True), 'updated_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'writers':
FieldInfo(annotation=Union[List, NoneType], required=True), 'year':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to [*FieldInfo*][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
original_title: Optional[str]
originally_available_at: Optional[str]
parent_guid: Optional[str]
parent_media_index: Optional[str]
parent_rating_key: Optional[str]
parent_thumb: Optional[str]
parent_title: Optional[str]
rating: Optional[str]
rating_image: Optional[str]
rating_key: Optional[str]
section_id: Optional[str]
sort_title: Optional[str]
studio: Optional[str]
summary: Optional[str]
tagline: Optional[str]
thumb: Optional[str]
title: Optional[str]
updated_at: Optional[str]
user_rating: Optional[str]
writers: Optional[List]
year: Optional[str]
```

```
class tautulli.models.search_results.ShowItem(**data)
```

```
    actors: Optional[List[str]]
    added_at: Optional[str]
    art: Optional[str]
    audience_rating: Optional[str]
    audience_rating_image: Optional[str]
    banner: Optional[str]
```

```
children_count: Optional[int]
collections: Optional[List]
content_rating: Optional[str]
directors: Optional[List]
duration: Optional[str]
full_title: Optional[str]
genres: Optional[List[str]]
grandparent_guid: Optional[str]
grandparent_rating_key: Optional[str]
grandparent_thumb: Optional[str]
grandparent_title: Optional[str]
guid: Optional[str]
guids: Optional[List]
labels: Optional[List]
last_viewed_at: Optional[str]
library_name: Optional[str]
live: Optional[int]
media_index: Optional[str]
media_info: Optional[List]
media_type: Optional[str]
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].
```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'actors':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'added_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'art':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audience_rating_image':
FieldInfo(annotation=Union[str, NoneType], required=True), 'banner':
FieldInfo(annotation=Union[str, NoneType], required=True), 'children_count':
FieldInfo(annotation=Union[int, NoneType], required=True), 'collections':
FieldInfo(annotation=Union[List, NoneType], required=True), 'content_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'directors':
FieldInfo(annotation=Union[List, NoneType], required=True), 'duration':
FieldInfo(annotation=Union[str, NoneType], required=True), 'full_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'genres':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'grandparent_guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_rating_key':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'grandparent_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guids':
FieldInfo(annotation=Union[List, NoneType], required=True), 'labels':
FieldInfo(annotation=Union[List, NoneType], required=True), 'last_viewed_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'library_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'live':
FieldInfo(annotation=Union[int, NoneType], required=True), 'media_index':
FieldInfo(annotation=Union[str, NoneType], required=True), 'media_info':
FieldInfo(annotation=Union[List, NoneType], required=True), 'media_type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'original_title':
FieldInfo(annotation=Union[str, NoneType], required=True),
'originally_available_at': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_guid': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_media_index': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_rating_key': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_thumb': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_image': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_key': FieldInfo(annotation=Union[str, NoneType],
required=True), 'section_id': FieldInfo(annotation=Union[str, NoneType],
required=True), 'sort_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'studio': FieldInfo(annotation=Union[str, NoneType],
required=True), 'summary': FieldInfo(annotation=Union[str, NoneType],
required=True), 'tagline': FieldInfo(annotation=Union[str, NoneType],
required=True), 'thumb': FieldInfo(annotation=Union[str, NoneType], required=True),
'title': FieldInfo(annotation=Union[str, NoneType], required=True), 'updated_at':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user_rating':
FieldInfo(annotation=Union[str, NoneType], required=True), 'writers':
FieldInfo(annotation=Union[List, NoneType], required=True), 'year':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

`original_title: Optional[str]`

```
originally_available_at: Optional[str]
parent_guid: Optional[str]
parent_media_index: Optional[str]
parent_rating_key: Optional[str]
parent_thumb: Optional[str]
parent_title: Optional[str]
rating: Optional[str]
rating_image: Optional[str]
rating_key: Optional[str]
section_id: Optional[str]
sort_title: Optional[str]
studio: Optional[str]
summary: Optional[str]
tagline: Optional[str]
thumb: Optional[str]
title: Optional[str]
updated_at: Optional[str]
user_rating: Optional[str]
writers: Optional[List]
year: Optional[str]
```

```
class tautulli.models.search_results.Stream(**data)
```

```
audio_bitrate: Optional[str]
audio_bitrate_mode: Optional[str]
audio_channel_layout: Optional[str]
audio_channels: Optional[str]
audio_codec: Optional[str]
audio_language: Optional[str]
audio_language_code: Optional[str]
audio_profile: Optional[str]
audio_sample_rate: Optional[str]
id: Optional[str]
```



```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'audio_bitrate':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_bitrate_mode':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_channel_layout':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_channels':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_codec':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_language':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_language_code':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_profile':
FieldInfo(annotation=Union[str, NoneType], required=False), 'audio_sample_rate':
FieldInfo(annotation=Union[str, NoneType], required=False), 'id':
FieldInfo(annotation=Union[str, NoneType], required=False), 'selected':
FieldInfo(annotation=Union[int, NoneType], required=True), 'subtitle_codec':
FieldInfo(annotation=Union[str, NoneType], required=False), 'subtitle_container':
FieldInfo(annotation=Union[str, NoneType], required=False), 'subtitle_forced':
FieldInfo(annotation=Union[int, NoneType], required=False), 'subtitle_format':
FieldInfo(annotation=Union[str, NoneType], required=False), 'subtitle_language':
FieldInfo(annotation=Union[str, NoneType], required=False),
'subtitle_language_code': FieldInfo(annotation=Union[str, NoneType],
required=False), 'subtitle_location': FieldInfo(annotation=Union[str, NoneType],
required=False), 'type': FieldInfo(annotation=Union[str, NoneType], required=True),
'video_bit_depth': FieldInfo(annotation=Union[str, NoneType], required=False),
'video_bitrate': FieldInfo(annotation=Union[str, NoneType], required=False),
'video_chroma_subsampling': FieldInfo(annotation=Union[str, NoneType],
required=False), 'video_codec': FieldInfo(annotation=Union[str, NoneType],
required=False), 'video_codec_level': FieldInfo(annotation=Union[str, NoneType],
required=False), 'video_color_primaries': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_color_range': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_color_space': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_color_trc': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_frame_rate': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_height': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_language': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_language_code': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_profile': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_ref_frames': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_scan_type': FieldInfo(annotation=Union[str,
NoneType], required=False), 'video_width': FieldInfo(annotation=Union[str,
NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
selected: Optional[int]
```

```
subtitle_codec: Optional[str]
```

```
subtitle_container: Optional[str]
```

```
subtitle_forced: Optional[int]
```

```
subtitle_format: Optional[str]
subtitle_language: Optional[str]
subtitle_language_code: Optional[str]
subtitle_location: Optional[str]
type: Optional[str]
video_bit_depth: Optional[str]
video_bitrate: Optional[str]
video_chroma_subsampling: Optional[str]
video_codec: Optional[str]
video_codec_level: Optional[str]
video_color_primaries: Optional[str]
video_color_range: Optional[str]
video_color_space: Optional[str]
video_color_trc: Optional[str]
video_frame_rate: Optional[str]
video_height: Optional[str]
video_language: Optional[str]
video_language_code: Optional[str]
video_profile: Optional[str]
video_ref_frames: Optional[str]
video_scan_type: Optional[str]
video_width: Optional[str]
```

2.3.32 Server ID

```
class tautulli.models.server_id.ServerID(**data)

    identifier: Optional[Any]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'identifier':
    FieldInfo(annotation=Union[Any, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.
```

2.3.33 Server Identity

```
class tautulli.models.server_identity.ServerIdentity(**data)

    machine_identifier: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'machine_identifier':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'version':
    FieldInfo(annotation=Union[str, NoneType], required=True)}
    Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

    version: Optional[str]
```

2.3.34 Server Info

```
class tautulli.models.server_info.ServerInfo(**data)

    model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'pms_identifier':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_ip':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_is_remote':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_name':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_platform':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_plexpass':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_port':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_ssl':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_url':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_url_manual':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_version':
    FieldInfo(annotation=Union[str, NoneType], required=True)}
    Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

    pms_identifier: Optional[str]

    pms_ip: Optional[str]

    pms_is_remote: Optional[int]

    pms_name: Optional[str]

    pms_platform: Optional[str]

    pms_plexpass: Optional[int]
```

```
pms_port: Optional[int]
pms_ssl: Optional[int]
pms_url: Optional[str]
pms_url_manual: Optional[int]
pms_version: Optional[str]
```

2.3.35 Server List

```
class tautulli.models.server_list.ServerListEntry(**data)

    clientIdentifier: Optional[str]

    httpsRequired: Optional[str]

    ip: Optional[str]

    is_cloud: Optional[bool]

    label: Optional[str]

    local: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'clientIdentifier':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'httpsRequired':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'ip':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'is_cloud':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'label':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'local':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'port':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'uri':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'value':
    FieldInfo(annotation=Union[str, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    port: Optional[str]

    uri: Optional[str]

    value: Optional[str]
```

2.3.36 Servers Info

```
class tautulli.models.servers_info.ServersInfoEntry(**data)

    host: Optional[str]

    machine_identifier: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'host':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'machine_identifier':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'name':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'port':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'version':
        FieldInfo(annotation=Union[str, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    name: Optional[str]

    port: Optional[str]

    version: Optional[str]
```

2.3.37 Server Status

```
class tautulli.models.server_status.ServerStatus(**data)

    connected: Optional[bool]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'connected':
        FieldInfo(annotation=Union[bool, NoneType], required=True), 'result':
        FieldInfo(annotation=Union[str, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    result: Optional[str]
```

2.3.38 Settings

```
class tautulli.models.settings.Advanced(**data)

    add_live_tv_library: Optional[bool]

    cache_size_mb: Optional[str]

    check_github_cache_seconds: Optional[str]

    config_version: Optional[str]

    export_threads: Optional[str]

    journal_mode: Optional[str]

    jwt_secret: Optional[str]

    jwt_update_secret: Optional[bool]

    metadata_cache_seconds: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'add_live_tv_library':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'cache_size_mb':
    FieldInfo(annotation=Union[str, NoneType], required=True),
    'check_github_cache_seconds': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'config_version': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'export_threads': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'journal_mode': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'jwt_secret': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'jwt_update_secret': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'metadata_cache_seconds': FieldInfo(annotation=Union[str,
    NoneType], required=True), 'notification_threads': FieldInfo(annotation=Union[str,
    NoneType], required=True), 'pms_timeout': FieldInfo(annotation=Union[str,
    NoneType], required=True), 'pms_update_check_interval':
    FieldInfo(annotation=Union[str, NoneType], required=True),
    'remote_access_ping_interval': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'remote_access_ping_threshold': FieldInfo(annotation=Union[str,
    NoneType], required=True), 'session_db_write_attempts':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'synchronous_mode':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'system_analytics':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'verbose_logs':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'verify_ssl_cert':
    FieldInfo(annotation=Union[bool, NoneType], required=True),
    'websocket_connection_attempts': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'websocket_connection_timeout': FieldInfo(annotation=Union[str,
    NoneType], required=True), 'websocket_monitor_ping_pong':
    FieldInfo(annotation=Union[bool, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to [Field-Info][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```

notification_threads: Optional[str]
pms_timeout: Optional[str]
pms_update_check_interval: Optional[str]
remote_access_ping_interval: Optional[str]
remote_access_ping_threshold: Optional[str]
session_db_write_attempts: Optional[str]
synchronous_mode: Optional[str]
system_analytics: Optional[bool]
verbose_logs: Optional[bool]
verify_ssl_cert: Optional[bool]
websocket_connection_attempts: Optional[str]
websocket_connection_timeout: Optional[str]
websocket_monitor_ping_pong: Optional[bool]
class tautulli.models.settings.Boxcar(**data)
    boxcar_enabled: Optional[bool]
    boxcar_on_buffer: Optional[bool]
    boxcar_on_concurrent: Optional[bool]
    boxcar_on_created: Optional[bool]
    boxcar_on_extdown: Optional[bool]
    boxcar_on_extup: Optional[bool]
    boxcar_on_intdown: Optional[bool]
    boxcar_on_intup: Optional[bool]
    boxcar_on_newdevice: Optional[bool]
    boxcar_on_pause: Optional[bool]
    boxcar_on_play: Optional[bool]
    boxcar_on_pmsupdate: Optional[bool]
    boxcar_on_resume: Optional[bool]
    boxcar_on_stop: Optional[bool]
    boxcar_on_watched: Optional[bool]
    boxcar_sound: Optional[str]
    boxcar_token: Optional[str]

```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'boxcar_enabled':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_buffer':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_concurrent':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_created':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_extdown':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_extup':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_intdown':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_intup':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_newdevice':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_pause':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_play':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_pmsupdate':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_resume':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_stop':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_on_watched':
```

```
FieldInfo(annotation=Union[bool, NoneType], required=True), 'boxcar_sound':
```

```
FieldInfo(annotation=Union[str, NoneType], required=True), 'boxcar_token':
```

```
FieldInfo(annotation=Union[str, NoneType], required=True)]
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
class tautulli.models.settings.Browser(**data)
```

```
    browser_auto_hide_delay: Optional[str]
```

```
    browser_enabled: Optional[bool]
```

```
    browser_on_buffer: Optional[bool]
```

```
    browser_on_concurrent: Optional[bool]
```

```
    browser_on_created: Optional[bool]
```

```
    browser_on_extdown: Optional[bool]
```

```
    browser_on_extup: Optional[bool]
```

```
    browser_on_intdown: Optional[bool]
```

```
    browser_on_intup: Optional[bool]
```

```
    browser_on_newdevice: Optional[bool]
```

```
    browser_on_pause: Optional[bool]
```

```
    browser_on_play: Optional[bool]
```

```
    browser_on_pmsupdate: Optional[bool]
```

```
    browser_on_resume: Optional[bool]
```

```
    browser_on_stop: Optional[bool]
```



```
browser_on_watched: Optional[bool]
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'browser_auto_hide_delay':
FieldInfo(annotation=Union[str, NoneType], required=True), 'browser_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_concurrent':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_created':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_extdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_extup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_intdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_intup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_newdevice':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_play':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'browser_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
class tautulli.models.settings.Cloudinary(**data)
```

```
cloudinary_api_key: Optional[str]
```

```
cloudinary_api_secret: Optional[str]
```

```
cloudinary_cloud_name: Optional[str]
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'cloudinary_api_key':
FieldInfo(annotation=Union[str, NoneType], required=True), 'cloudinary_api_secret':
FieldInfo(annotation=Union[str, NoneType], required=True), 'cloudinary_cloud_name':
FieldInfo(annotation=Union[str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
class tautulli.models.settings.Email(**data)
```

```
email_bcc: Optional[str]
```

```
email_cc: Optional[str]
```

```
email_enabled: Optional[bool]
```

```
email_from: Optional[str]
email_from_name: Optional[str]
email_html_support: Optional[bool]
email_on_buffer: Optional[bool]
email_on_concurrent: Optional[bool]
email_on_created: Optional[bool]
email_on_extdown: Optional[bool]
email_on_extup: Optional[bool]
email_on_intdown: Optional[bool]
email_on_intup: Optional[bool]
email_on_newdevice: Optional[bool]
email_on_pause: Optional[bool]
email_on_play: Optional[bool]
email_on_pmsupdate: Optional[bool]
email_on_resume: Optional[bool]
email_on_stop: Optional[bool]
email_on_watched: Optional[bool]
email_smtp_password: Optional[str]
email_smtp_port: Optional[str]
email_smtp_server: Optional[str]
email_smtp_user: Optional[str]
email_tls: Optional[bool]
email_to: Optional[str]
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].
```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'email_bcc':
FieldInfo(annotation=Union[str, NoneType], required=True), 'email_cc':
FieldInfo(annotation=Union[str, NoneType], required=True), 'email_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_from':
FieldInfo(annotation=Union[str, NoneType], required=True), 'email_from_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'email_html_support':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_concurrent':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_created':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_extdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_extup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_intdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_intup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_newdevice':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_play':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_smtp_password':
FieldInfo(annotation=Union[str, NoneType], required=True), 'email_smtp_port':
FieldInfo(annotation=Union[str, NoneType], required=True), 'email_smtp_server':
FieldInfo(annotation=Union[str, NoneType], required=True), 'email_smtp_user':
FieldInfo(annotation=Union[str, NoneType], required=True), 'email_tls':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'email_to':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
class tautulli.models.settings.Facebook(**data)
```

```

    facebook_app_id: Optional[str]

    facebook_app_secret: Optional[str]

    facebook_enabled: Optional[bool]

    facebook_group: Optional[str]

    facebook_incl_pmslink: Optional[bool]

    facebook_incl_poster: Optional[bool]

    facebook_incl_subject: Optional[bool]

    facebook_on_buffer: Optional[bool]

    facebook_on_concurrent: Optional[bool]

    facebook_on_created: Optional[bool]

    facebook_on_extdown: Optional[bool]

    facebook_on_extup: Optional[bool]

```

```
facebook_on_intdown: Optional[bool]

facebook_on_intup: Optional[bool]

facebook_on_newdevice: Optional[bool]

facebook_on_pause: Optional[bool]

facebook_on_play: Optional[bool]

facebook_on_pmsupdate: Optional[bool]

facebook_on_resume: Optional[bool]

facebook_on_stop: Optional[bool]

facebook_on_watched: Optional[bool]

facebook_redirect_uri: Optional[str]

facebook_token: Optional[str]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'facebook_app_id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'facebook_app_secret':
FieldInfo(annotation=Union[str, NoneType], required=True), 'facebook_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'facebook_group':
FieldInfo(annotation=Union[str, NoneType], required=True), 'facebook_incl_pmslink':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'facebook_incl_poster':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'facebook_incl_subject':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'facebook_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True),
'facebook_on_concurrent': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'facebook_on_created': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'facebook_on_extdown': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'facebook_on_extup': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'facebook_on_intdown': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'facebook_on_intup': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'facebook_on_newdevice': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'facebook_on_pause': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'facebook_on_play': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'facebook_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'facebook_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'facebook_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'facebook_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'facebook_redirect_uri':
FieldInfo(annotation=Union[str, NoneType], required=True), 'facebook_token':
FieldInfo(annotation=Union[str, NoneType], required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

class tautulli.models.settings.General(**data)
```

```
allow_guest_access: Optional[bool]
anon_redirect: Optional[str]
api_enabled: Optional[bool]
api_key: Optional[str]
api_sql: Optional[bool]
backup_days: Optional[str]
backup_dir: Optional[str]
backup_interval: Optional[str]
cache_dir: Optional[str]
cache_images: Optional[bool]
check_github: Optional[bool]
check_github_interval: Optional[str]
check_github_on_startup: Optional[bool]
cleanup_files: Optional[bool]
config_version: Optional[str]
date_format: Optional[str]
do_not_override_git_branch: Optional[bool]
enable_https: Optional[bool]
export_dir: Optional[str]
first_run_complete: Optional[bool]
freeze_db: Optional[bool]
geoip_db: Optional[str]
geoip_db_installed: Optional[bool]
geoip_db_update_days: Optional[str]
get_file_sizes: Optional[bool]
get_file_sizes_hold: Optional[GetFileSizesHold]
git_branch: Optional[str]
git_path: Optional[str]
git_remote: Optional[str]
git_repo: Optional[str]
git_token: Optional[str]
```

```
git_user: Optional[str]
graph_days: Optional[str]
graph_months: Optional[str]
graph_tab: Optional[str]
graph_type: Optional[str]
group_history_tables: Optional[bool]
history_table_activity: Optional[bool]
home_library_cards: Optional[List[str]]
home_refresh_interval: Optional[str]
home_sections: Optional[List[str]]
home_stats_cards: Optional[List[str]]
home_stats_count: Optional[str]
home_stats_length: Optional[str]
home_stats_recently_added_count: Optional[str]
home_stats_type: Optional[bool]
http_base_url: Optional[str]
http_basic_auth: Optional[bool]
http_environment: Optional[str]
http_hash_password: Optional[bool]
http_hashed_password: Optional[bool]
http_host: Optional[str]
http_password: Optional[str]
http_plex_admin: Optional[bool]
http_port: Optional[str]
http_proxy: Optional[bool]
http_rate_limit_attempts: Optional[str]
http_rate_limit_attempts_interval: Optional[str]
http_rate_limit_lockout_time: Optional[str]
http_root: Optional[str]
http_username: Optional[str]
https_cert: Optional[str]
```

```
https_cert_chain: Optional[str]
https_create_cert: Optional[bool]
https_domain: Optional[str]
https_ip: Optional[str]
https_key: Optional[str]
interface: Optional[str]
interface_list: Optional[List[str]]
ip_logging_enable: Optional[bool]
launch_browser: Optional[bool]
launch_startup: Optional[bool]
log_blacklist: Optional[bool]
log_dir: Optional[str]
maxmind_license_key: Optional[str]
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].
```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'allow_guest_access':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'anon_redirect':
FieldInfo(annotation=Union[str, NoneType], required=True), 'api_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'api_key':
FieldInfo(annotation=Union[str, NoneType], required=True), 'api_sql':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'backup_days':
FieldInfo(annotation=Union[str, NoneType], required=True), 'backup_dir':
FieldInfo(annotation=Union[str, NoneType], required=True), 'backup_interval':
FieldInfo(annotation=Union[str, NoneType], required=True), 'cache_dir':
FieldInfo(annotation=Union[str, NoneType], required=True), 'cache_images':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'check_github':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'check_github_interval':
FieldInfo(annotation=Union[str, NoneType], required=True),
'check_github_on_startup': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'cleanup_files': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'config_version': FieldInfo(annotation=Union[str, NoneType],
required=True), 'date_format': FieldInfo(annotation=Union[str, NoneType],
required=True), 'do_not_override_git_branch': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'enable_https': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'export_dir': FieldInfo(annotation=Union[str, NoneType],
required=True), 'first_run_complete': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'freeze_db': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'geop_db': FieldInfo(annotation=Union[str, NoneType],
required=True), 'geop_db_installed': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'geop_db_update_days': FieldInfo(annotation=Union[str, NoneType],
required=True), 'get_file_sizes': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'get_file_sizes_hold': FieldInfo(annotation=Union[GetFileSizesHold,
NoneType], required=True), 'git_branch': FieldInfo(annotation=Union[str, NoneType],
required=True), 'git_path': FieldInfo(annotation=Union[str, NoneType],
required=True), 'git_remote': FieldInfo(annotation=Union[str, NoneType],
required=True), 'git_repo': FieldInfo(annotation=Union[str, NoneType],
required=True), 'git_token': FieldInfo(annotation=Union[str, NoneType],
required=True), 'git_user': FieldInfo(annotation=Union[str, NoneType],
required=True), 'graph_days': FieldInfo(annotation=Union[str, NoneType],
required=True), 'graph_months': FieldInfo(annotation=Union[str, NoneType],
required=True), 'graph_tab': FieldInfo(annotation=Union[str, NoneType],
required=True), 'graph_type': FieldInfo(annotation=Union[str, NoneType],
required=True), 'group_history_tables': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'history_table_activity': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'home_library_cards':
FieldInfo(annotation=Union[List[str], NoneType], required=True),
'home_refresh_interval': FieldInfo(annotation=Union[str, NoneType], required=True),
'home_sections': FieldInfo(annotation=Union[List[str], NoneType], required=True),
'home_stats_cards': FieldInfo(annotation=Union[List[str], NoneType],
required=True), 'home_stats_count': FieldInfo(annotation=Union[str, NoneType],
required=True), 'home_stats_length': FieldInfo(annotation=Union[str, NoneType],
required=True), 'home_stats_recently_added_count': FieldInfo(annotation=Union[str,
NoneType], required=True), 'home_stats_type': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'http_base_url': FieldInfo(annotation=Union[str,
NoneType], required=True), 'http_basic_auth': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'http_environment': FieldInfo(annotation=Union[str,
NoneType], required=True), 'http_hash_password': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'http_hashed_password': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'http_host': FieldInfo(annotation=Union[str, NoneType],
required=True), 'http_password': FieldInfo(annotation=Union[str, NoneType],
required=True), 'http_plex_admin': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'http_port': FieldInfo(annotation=Union[str, NoneType],
required=True), 'http_proxy': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'http_rate_limit_attempts': FieldInfo(annotation=Union[str,
NoneType], required=True), 'http_rate_limit_attempts_interval':

```


Metadata about the fields defined on the model, mapping of field names to [*FieldInfo*][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
musicbrainz_lookup: Optional[bool]
plexpy_auto_update: Optional[bool]
show_advanced_settings: Optional[bool]
sys_tray_icon: Optional[bool]
themoviedb_apikey: Optional[str]
themoviedb_lookup: Optional[bool]
time_format: Optional[str]
tvmaze_lookup: Optional[bool]
update_db_interval: Optional[str]
update_labels: Optional[bool]
update_libraries_db_notify: Optional[bool]
update_notifiers_db: Optional[bool]
update_section_ids: Optional[bool]
update_show_changelog: Optional[int]
week_start_monday: Optional[bool]
win_sys_tray: Optional[bool]
```

```
class tautulli.models.settings.GetFilesizesHold(**data)
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to [*ConfigDict*][pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'rating_keys':
FieldInfo(annotation=Union[List, NoneType], required=True), 'section_ids':
FieldInfo(annotation=Union[List, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to [*FieldInfo*][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
rating_keys: Optional[List]
section_ids: Optional[List]
```

```
class tautulli.models.settings.Growl(**data)
```

```
growl_enabled: Optional[bool]
```

```
growl_host: Optional[str]
growl_on_buffer: Optional[bool]
growl_on_concurrent: Optional[bool]
growl_on_created: Optional[bool]
growl_on_extdown: Optional[bool]
growl_on_extup: Optional[bool]
growl_on_intdown: Optional[bool]
growl_on_intup: Optional[bool]
growl_on_newdevice: Optional[bool]
growl_on_pause: Optional[bool]
growl_on_play: Optional[bool]
growl_on_pmsupdate: Optional[bool]
growl_on_resume: Optional[bool]
growl_on_stop: Optional[bool]
growl_on_watched: Optional[bool]
growl_password: Optional[str]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'growl_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_host':
FieldInfo(annotation=Union[str, NoneType], required=True), 'growl_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_concurrent':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_created':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_extdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_extup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_intdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_intup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_newdevice':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_play':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'growl_password':
FieldInfo(annotation=Union[str, NoneType], required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.
```

```
class tautulli.models.settings.Hipchat(**data)

    hipchat_color: Optional[str]
    hipchat_emoticon: Optional[str]
    hipchat_enabled: Optional[bool]
    hipchat_incl_pmslink: Optional[bool]
    hipchat_incl_poster: Optional[bool]
    hipchat_incl_subject: Optional[bool]
    hipchat_on_buffer: Optional[bool]
    hipchat_on_concurrent: Optional[bool]
    hipchat_on_created: Optional[bool]
    hipchat_on_extdown: Optional[bool]
    hipchat_on_extup: Optional[bool]
    hipchat_on_intdown: Optional[bool]
    hipchat_on_intup: Optional[bool]
    hipchat_on_newdevice: Optional[bool]
    hipchat_on_pause: Optional[bool]
    hipchat_on_play: Optional[bool]
    hipchat_on_pmsupdate: Optional[bool]
    hipchat_on_resume: Optional[bool]
    hipchat_on_stop: Optional[bool]
    hipchat_on_watched: Optional[bool]
    hipchat_url: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].
```

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'hipchat_color':
FieldInfo(annotation=Union[str, NoneType], required=True), 'hipchat_emoticon':
FieldInfo(annotation=Union[str, NoneType], required=True), 'hipchat_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_incl_pmslink':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_incl_poster':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_incl_subject':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_concurrent':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_created':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_extdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_extup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_intdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_intup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_newdevice':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_play':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'hipchat_url':
FieldInfo(annotation=Union[str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
class tautulli.models.settings.IFTTT(**data)
```

```
    ifttt_enabled: Optional[bool]

    ifttt_event: Optional[str]

    ifttt_key: Optional[str]

    ifttt_on_buffer: Optional[bool]

    ifttt_on_concurrent: Optional[bool]

    ifttt_on_created: Optional[bool]

    ifttt_on_extdown: Optional[bool]

    ifttt_on_extup: Optional[bool]

    ifttt_on_intdown: Optional[bool]

    ifttt_on_intup: Optional[bool]

    ifttt_on_newdevice: Optional[bool]

    ifttt_on_pause: Optional[bool]

    ifttt_on_play: Optional[bool]

    ifttt_on_pmsupdate: Optional[bool]

    ifttt_on_resume: Optional[bool]
```

```

ifttt_on_stop: Optional[bool]

ifttt_on_watched: Optional[bool]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'ifttt_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_event':
FieldInfo(annotation=Union[str, NoneType], required=True), 'ifttt_key':
FieldInfo(annotation=Union[str, NoneType], required=True), 'ifttt_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_concurrent':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_created':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_extdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_extup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_intdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_intup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_newdevice':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_play':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'ifttt_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

```

This replaces *Model.__fields__* from Pydantic V1.

```

class tautulli.models.settings.Join(**data)

    join_apikey: Optional[str]

    join_deviceid: Optional[str]

    join_enabled: Optional[bool]

    join_incl_subject: Optional[bool]

    join_on_buffer: Optional[bool]

    join_on_concurrent: Optional[bool]

    join_on_created: Optional[bool]

    join_on_extdown: Optional[bool]

    join_on_extup: Optional[bool]

    join_on_intdown: Optional[bool]

    join_on_intup: Optional[bool]

    join_on_newdevice: Optional[bool]

    join_on_pause: Optional[bool]

```

```
join_on_play: Optional[bool]

join_on_pmsupdate: Optional[bool]

join_on_resume: Optional[bool]

join_on_stop: Optional[bool]

join_on_watched: Optional[bool]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'join_apikey':
FieldInfo(annotation=Union[str, NoneType], required=True), 'join_deviceid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'join_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_incl_subject':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_concurrent':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_created':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_extdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_extup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_intdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_intup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_newdevice':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_play':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'join_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].
```

This replaces *Model.__fields__* from Pydantic V1.

```
class tautulli.models.settings.Monitoring(**data)

    buffer_threshold: Optional[str]

    buffer_wait: Optional[str]

    imgur_client_id: Optional[str]

    logging_ignore_interval: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].
```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'buffer_threshold':
FieldInfo(annotation=Union[str, NoneType], required=True), 'buffer_wait':
FieldInfo(annotation=Union[str, NoneType], required=True), 'imgur_client_id':
FieldInfo(annotation=Union[str, NoneType], required=True),
'logging_ignore_interval': FieldInfo(annotation=Union[str, NoneType],
required=True), 'monitor_pms_updates': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'monitor_remote_access': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'monitoring_interval': FieldInfo(annotation=Union[str,
NoneType], required=True), 'monitoring_use_websocket':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'movie_logging_enable':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'movie_notify_enable':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'movie_notify_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'movie_notify_on_start':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'movie_notify_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'movie_watched_percent':
FieldInfo(annotation=Union[str, NoneType], required=True), 'music_logging_enable':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'music_notify_enable':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'music_notify_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'music_notify_on_start':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'music_notify_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'music_watched_percent':
FieldInfo(annotation=Union[str, NoneType], required=True),
'notify_concurrent_by_ip': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'notify_concurrent_threshold': FieldInfo(annotation=Union[str,
NoneType], required=True), 'notify_consecutive': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'notify_continued_session_threshold':
FieldInfo(annotation=Union[str, NoneType], required=True),
'notify_group_recently_added': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'notify_group_recently_added_grandparent':
FieldInfo(annotation=Union[bool, NoneType], required=True),
'notify_group_recently_added_parent': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'notify_new_device_initial_only': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'notify_on_buffer_body_text':
FieldInfo(annotation=Union[str, NoneType], required=True),
'notify_on_buffer_subject_text': FieldInfo(annotation=Union[str, NoneType],
required=True), 'notify_on_concurrent_body_text': FieldInfo(annotation=Union[str,
NoneType], required=True), 'notify_on_concurrent_subject_text':
FieldInfo(annotation=Union[str, NoneType], required=True),
'notify_on_created_body_text': FieldInfo(annotation=Union[str, NoneType],
required=True), 'notify_on_created_subject_text': FieldInfo(annotation=Union[str,
NoneType], required=True), 'notify_on_extdown_body_text':
FieldInfo(annotation=Union[str, NoneType], required=True),
'notify_on_extdown_subject_text': FieldInfo(annotation=Union[str, NoneType],
required=True), 'notify_on_extup_body_text': FieldInfo(annotation=Union[str,
NoneType], required=True), 'notify_on_extup_subject_text':
FieldInfo(annotation=Union[str, NoneType], required=True),
'notify_on_intdown_body_text': FieldInfo(annotation=Union[str, NoneType],
required=True), 'notify_on_intdown_subject_text': FieldInfo(annotation=Union[str,
NoneType], required=True), 'notify_on_intup_body_text':
FieldInfo(annotation=Union[str, NoneType], required=True),
'notify_on_intup_subject_text': FieldInfo(annotation=Union[str, NoneType],
required=True), 'notify_on_newdevice_body_text': FieldInfo(annotation=Union[str,
NoneType], required=True), 'notify_on_newdevice_subject_text':
FieldInfo(annotation=Union[str, NoneType], required=True),
'notify_on_pause_body_text': FieldInfo(annotation=Union[str, NoneType],
required=True), 'notify_on_pause_subject_text': FieldInfo(annotation=Union[str,
NoneType], required=True), 'notify_on_pmsupdate_body_text':
FieldInfo(annotation=Union[str, NoneType], required=True),
'notify_on_pmsupdate_subject_text': FieldInfo(annotation=Union[str, NoneType],
required=True), 'notify_on_resume_body_text': FieldInfo(annotation=Union[str,

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
monitor_pms_updates: Optional[bool]
monitor_remote_access: Optional[bool]
monitoring_interval: Optional[str]
monitoring_use_websocket: Optional[bool]
movie_logging_enable: Optional[bool]
movie_notify_enable: Optional[bool]
movie_notify_on_pause: Optional[bool]
movie_notify_on_start: Optional[bool]
movie_notify_on_stop: Optional[bool]
movie_watched_percent: Optional[str]
music_logging_enable: Optional[bool]
music_notify_enable: Optional[bool]
music_notify_on_pause: Optional[bool]
music_notify_on_start: Optional[bool]
music_notify_on_stop: Optional[bool]
music_watched_percent: Optional[str]
notify_concurrent_by_ip: Optional[bool]
notify_concurrent_threshold: Optional[str]
notify_consecutive: Optional[bool]
notify_continued_session_threshold: Optional[str]
notify_group_recently_added: Optional[bool]
notify_group_recently_added_grandparent: Optional[bool]
notify_group_recently_added_parent: Optional[bool]
notify_new_device_initial_only: Optional[bool]
notify_on_buffer_body_text: Optional[str]
notify_on_buffer_subject_text: Optional[str]
notify_on_concurrent_body_text: Optional[str]
notify_on_concurrent_subject_text: Optional[str]
```


notify_on_created_body_text: Optional[str]
notify_on_created_subject_text: Optional[str]
notify_on_extdown_body_text: Optional[str]
notify_on_extdown_subject_text: Optional[str]
notify_on_extup_body_text: Optional[str]
notify_on_extup_subject_text: Optional[str]
notify_on_intdown_body_text: Optional[str]
notify_on_intdown_subject_text: Optional[str]
notify_on_intup_body_text: Optional[str]
notify_on_intup_subject_text: Optional[str]
notify_on_newdevice_body_text: Optional[str]
notify_on_newdevice_subject_text: Optional[str]
notify_on_pause_body_text: Optional[str]
notify_on_pause_subject_text: Optional[str]
notify_on_pmsupdate_body_text: Optional[str]
notify_on_pmsupdate_subject_text: Optional[str]
notify_on_resume_body_text: Optional[str]
notify_on_resume_subject_text: Optional[str]
notify_on_start_body_text: Optional[str]
notify_on_start_subject_text: Optional[str]
notify_on_stop_body_text: Optional[str]
notify_on_stop_subject_text: Optional[str]
notify_on_watched_body_text: Optional[str]
notify_on_watched_subject_text: Optional[str]
notify_recently_added: Optional[bool]
notify_recently_added_delay: Optional[str]
notify_recently_added_grandparent: Optional[bool]
notify_recently_added_upgrade: Optional[bool]
notify_remote_access_threshold: Optional[str]
notify_scripts_args_text: Optional[str]
notify_upload_posters: Optional[str]

```
notify_watched_percent: Optional[str]
refresh_libraries_interval: Optional[str]
refresh_libraries_on_startup: Optional[bool]
refresh_users_interval: Optional[str]
refresh_users_on_startup: Optional[bool]
session_db_write_attempts: Optional[str]
tv_logging_enable: Optional[bool]
tv_notify_enable: Optional[bool]
tv_notify_on_pause: Optional[bool]
tv_notify_on_start: Optional[bool]
tv_notify_on_stop: Optional[bool]
tv_watched_percent: Optional[str]
video_logging_enable: Optional[bool]
```

```
class tautulli.models.settings.NMA(**data)
```

```
    model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
    model_fields: ClassVar[dict[str, FieldInfo]] = {'nma_apikey':
FieldInfo(annotation=Union[str, NoneType], required=True), 'nma_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_concurrent':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_created':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_extdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_extup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_intdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_intup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_newdevice':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_play':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'nma_priority':
FieldInfo(annotation=Union[bool, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
nma_apikey: Optional[str]
nma_enabled: Optional[bool]
```

```

nma_on_buffer: Optional[bool]
nma_on_concurrent: Optional[bool]
nma_on_created: Optional[bool]
nma_on_extdown: Optional[bool]
nma_on_extup: Optional[bool]
nma_on_intdown: Optional[bool]
nma_on_intup: Optional[bool]
nma_on_newdevice: Optional[bool]
nma_on_pause: Optional[bool]
nma_on_play: Optional[bool]
nma_on_pmsupdate: Optional[bool]
nma_on_resume: Optional[bool]
nma_on_stop: Optional[bool]
nma_on_watched: Optional[bool]
nma_priority: Optional[bool]

```

```
class tautulli.models.settings.Newsletter(**data)
```

```
    model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'newsletter_auth':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'newsletter_custom_dir':
FieldInfo(annotation=Union[str, NoneType], required=True), 'newsletter_dir':
FieldInfo(annotation=Union[str, NoneType], required=True),
'newsletter_inline_styles': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'newsletter_password': FieldInfo(annotation=Union[str, NoneType],
required=True), 'newsletter_self_hosted': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'newsletter_static_url':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'newsletter_templates':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```

newsletter_auth: Optional[bool]
newsletter_custom_dir: Optional[str]
newsletter_dir: Optional[str]
newsletter_inline_styles: Optional[bool]

```

```
newsletter_password: Optional[str]

newsletter_self_hosted: Optional[bool]

newsletter_static_url: Optional[bool]

newsletter_templates: Optional[str]

class tautulli.models.settings.OSXNotify(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'osx_notify_app':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'osx_notify_enabled':
        FieldInfo(annotation=Union[bool, NoneType], required=True), 'osx_notify_on_buffer':
        FieldInfo(annotation=Union[bool, NoneType], required=True),
        'osx_notify_on_concurrent': FieldInfo(annotation=Union[bool, NoneType],
        required=True), 'osx_notify_on_created': FieldInfo(annotation=Union[bool,
        NoneType], required=True), 'osx_notify_on_extdown':
        FieldInfo(annotation=Union[bool, NoneType], required=True), 'osx_notify_on_extup':
        FieldInfo(annotation=Union[bool, NoneType], required=True), 'osx_notify_on_intdown':
        FieldInfo(annotation=Union[bool, NoneType], required=True), 'osx_notify_on_intup':
        FieldInfo(annotation=Union[bool, NoneType], required=True),
        'osx_notify_on_newdevice': FieldInfo(annotation=Union[bool, NoneType],
        required=True), 'osx_notify_on_pause': FieldInfo(annotation=Union[bool, NoneType],
        required=True), 'osx_notify_on_play': FieldInfo(annotation=Union[bool, NoneType],
        required=True), 'osx_notify_on_pmsupdate': FieldInfo(annotation=Union[bool,
        NoneType], required=True), 'osx_notify_on_resume': FieldInfo(annotation=Union[bool,
        NoneType], required=True), 'osx_notify_on_stop': FieldInfo(annotation=Union[bool,
        NoneType], required=True), 'osx_notify_on_watched':
        FieldInfo(annotation=Union[bool, NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    osx_notify_app: Optional[str]

    osx_notify_enabled: Optional[bool]

    osx_notify_on_buffer: Optional[bool]

    osx_notify_on_concurrent: Optional[bool]

    osx_notify_on_created: Optional[bool]

    osx_notify_on_extdown: Optional[bool]

    osx_notify_on_extup: Optional[bool]

    osx_notify_on_intdown: Optional[bool]

    osx_notify_on_intup: Optional[bool]

    osx_notify_on_newdevice: Optional[bool]
```

```

osx_notify_on_pause: Optional[bool]

osx_notify_on_play: Optional[bool]

osx_notify_on_pmsupdate: Optional[bool]

osx_notify_on_resume: Optional[bool]

osx_notify_on_stop: Optional[bool]

osx_notify_on_watched: Optional[bool]

class tautulli.models.settings.PMS(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'pms_identifier':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_ip':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_is_cloud':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_is_remote':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_logs_folder':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_logs_line_cap':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_name':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_platform':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_plexpass':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_port':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_ssl':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'pms_token':
    FieldInfo(annotation=Union[str, NoneType], required=False), 'pms_update_channel':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pms_update_distro':
    FieldInfo(annotation=Union[str, NoneType], required=True),
    'pms_update_distro_build': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'pms_url': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'pms_url_manual': FieldInfo(annotation=Union[int, NoneType],
    required=True), 'pms_url_override': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'pms_use_bif': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'pms_uuid': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'pms_version': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'pms_web_url': FieldInfo(annotation=Union[str, NoneType],
    required=True)}

        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    pms_identifier: Optional[str]

    pms_ip: Optional[str]

    pms_is_cloud: Optional[int]

    pms_is_remote: Optional[int]

    pms_logs_folder: Optional[str]

```

```
pms_logs_line_cap: Optional[str]
pms_name: Optional[str]
pms_platform: Optional[str]
pms_plexpass: Optional[int]
pms_port: Optional[int]
pms_ssl: Optional[int]
pms_token: Optional[str]
pms_update_channel: Optional[str]
pms_update_distro: Optional[str]
pms_update_distro_build: Optional[str]
pms_url: Optional[str]
pms_url_manual: Optional[int]
pms_url_override: Optional[str]
pms_use_bif: Optional[bool]
pms_uuid: Optional[str]
pms_version: Optional[str]
pms_web_url: Optional[str]
```

```
class tautulli.models.settings.Plex(**data)
```

```
    model_config: ClassVar[ConfigDict] = {}
```

```
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].
```

```
    model_fields: ClassVar[dict[str, FieldInfo]] = {'plex_client_host':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'plex_enabled':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_buffer':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_concurrent':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_created':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_extdown':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_extup':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_intdown':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_intup':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_newdevice':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_pause':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_play':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_pmsupdate':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_resume':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_stop':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_on_watched':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'plex_password':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'plex_username':
    FieldInfo(annotation=Union[str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
plex_client_host: Optional[str]
plex_enabled: Optional[bool]
plex_on_buffer: Optional[bool]
plex_on_concurrent: Optional[bool]
plex_on_created: Optional[bool]
plex_on_extdown: Optional[bool]
plex_on_extup: Optional[bool]
plex_on_intdown: Optional[bool]
plex_on_intup: Optional[bool]
plex_on_newdevice: Optional[bool]
plex_on_pause: Optional[bool]
plex_on_play: Optional[bool]
plex_on_pmsupdate: Optional[bool]
plex_on_resume: Optional[bool]
plex_on_stop: Optional[bool]
plex_on_watched: Optional[bool]
plex_password: Optional[str]
plex_username: Optional[str]
```

```
class tautulli.models.settings.PlexWatch(**data)
```

```
    grouping_charts: Optional[bool]
    grouping_global_history: Optional[bool]
    grouping_user_history: Optional[bool]
```

```
    model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
    model_fields: ClassVar[dict[str, FieldInfo]] = {'grouping_charts':
    FieldInfo(annotation=Union[bool, NoneType], required=True),
    'grouping_global_history': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'grouping_user_history': FieldInfo(annotation=Union[bool,
    NoneType], required=True), 'plexwatch_database': FieldInfo(annotation=Union[str,
    NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

plexwatch_database: Optional[str]

class tautulli.models.settings.Prowl(data)**

model_config: ClassVar[ConfigDict] = {}

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

**model_fields: ClassVar[dict[str, FieldInfo]] = {'prowl_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_keys':
FieldInfo(annotation=Union[str, NoneType], required=True), 'prowl_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_concurrent':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_created':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_extdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_extup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_intdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_intup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_newdevice':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_play':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'prowl_priority':
FieldInfo(annotation=Union[bool, NoneType], required=True)}**

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

prowl_enabled: Optional[bool]

prowl_keys: Optional[str]

prowl_on_buffer: Optional[bool]

prowl_on_concurrent: Optional[bool]

prowl_on_created: Optional[bool]

prowl_on_extdown: Optional[bool]

prowl_on_extup: Optional[bool]

prowl_on_intdown: Optional[bool]

prowl_on_intup: Optional[bool]

prowl_on_newdevice: Optional[bool]

prowl_on_pause: Optional[bool]


```

    prowl_on_play: Optional[bool]

    prowl_on_pmsupdate: Optional[bool]

    prowl_on_resume: Optional[bool]

    prowl_on_stop: Optional[bool]

    prowl_on_watched: Optional[bool]

    prowl_priority: Optional[bool]

class tautulli.models.settings.PushBullet(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'pushbullet_apikey':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pushbullet_channel_tag':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pushbullet_deviceid':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pushbullet_enabled':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushbullet_on_buffer':
    FieldInfo(annotation=Union[bool, NoneType], required=True),
    'pushbullet_on_concurrent': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'pushbullet_on_created': FieldInfo(annotation=Union[bool,
    NoneType], required=True), 'pushbullet_on_extdown':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushbullet_on_extup':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushbullet_on_intdown':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushbullet_on_intup':
    FieldInfo(annotation=Union[bool, NoneType], required=True),
    'pushbullet_on_newdevice': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'pushbullet_on_pause': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'pushbullet_on_play': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'pushbullet_on_pmsupdate': FieldInfo(annotation=Union[bool,
    NoneType], required=True), 'pushbullet_on_resume': FieldInfo(annotation=Union[bool,
    NoneType], required=True), 'pushbullet_on_stop': FieldInfo(annotation=Union[bool,
    NoneType], required=True), 'pushbullet_on_watched':
    FieldInfo(annotation=Union[bool, NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    pushbullet_apikey: Optional[str]

    pushbullet_channel_tag: Optional[str]

    pushbullet_deviceid: Optional[str]

    pushbullet_enabled: Optional[bool]

    pushbullet_on_buffer: Optional[bool]

    pushbullet_on_concurrent: Optional[bool]

    pushbullet_on_created: Optional[bool]

```

```
pushbullet_on_extdown: Optional[bool]
pushbullet_on_extup: Optional[bool]
pushbullet_on_intdown: Optional[bool]
pushbullet_on_intup: Optional[bool]
pushbullet_on_newdevice: Optional[bool]
pushbullet_on_pause: Optional[bool]
pushbullet_on_play: Optional[bool]
pushbullet_on_pmsupdate: Optional[bool]
pushbullet_on_resume: Optional[bool]
pushbullet_on_stop: Optional[bool]
pushbullet_on_watched: Optional[bool]
```

```
class tautulli.models.settings.Pushalot(**data)
```

```
    model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
    model_fields: ClassVar[dict[str, FieldInfo]] = {'pushalot_apikey':
FieldInfo(annotation=Union[str, NoneType], required=True), 'pushalot_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushalot_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True),
'pushalot_on_concurrent': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'pushalot_on_created': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'pushalot_on_extdown': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'pushalot_on_extup': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'pushalot_on_intdown': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'pushalot_on_intup': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'pushalot_on_newdevice': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'pushalot_on_pause': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'pushalot_on_play': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'pushalot_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushalot_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushalot_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushalot_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
pushalot_apikey: Optional[str]
pushalot_enabled: Optional[bool]
pushalot_on_buffer: Optional[bool]
pushalot_on_concurrent: Optional[bool]
```

```

pushalot_on_created: Optional[bool]
pushalot_on_extdown: Optional[bool]
pushalot_on_extup: Optional[bool]
pushalot_on_intdown: Optional[bool]
pushalot_on_intup: Optional[bool]
pushalot_on_newdevice: Optional[bool]
pushalot_on_pause: Optional[bool]
pushalot_on_play: Optional[bool]
pushalot_on_pmsupdate: Optional[bool]
pushalot_on_resume: Optional[bool]
pushalot_on_stop: Optional[bool]
pushalot_on_watched: Optional[bool]

class tautulli.models.settings.Pushover(**data)
    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'pushover_apitoken':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pushover_enabled':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushover_html_support':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushover_incl_pmslink':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushover_incl_url':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushover_keys':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'pushover_on_buffer':
    FieldInfo(annotation=Union[bool, NoneType], required=True),
    'pushover_on_concurrent': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'pushover_on_created': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'pushover_on_extdown': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'pushover_on_extup': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'pushover_on_intdown': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'pushover_on_intup': FieldInfo(annotation=Union[bool, NoneType],
    required=True), 'pushover_on_newdevice': FieldInfo(annotation=Union[bool,
    NoneType], required=True), 'pushover_on_pause': FieldInfo(annotation=Union[bool,
    NoneType], required=True), 'pushover_on_play': FieldInfo(annotation=Union[bool,
    NoneType], required=True), 'pushover_on_pmsupdate':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushover_on_resume':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushover_on_stop':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushover_on_watched':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushover_priority':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'pushover_sound':
    FieldInfo(annotation=Union[str, NoneType], required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

```

```
pushover_apitoken: Optional[str]
pushover_enabled: Optional[bool]
pushover_html_support: Optional[bool]
pushover_incl_pmslink: Optional[bool]
pushover_incl_url: Optional[bool]
pushover_keys: Optional[str]
pushover_on_buffer: Optional[bool]
pushover_on_concurrent: Optional[bool]
pushover_on_created: Optional[bool]
pushover_on_extdown: Optional[bool]
pushover_on_extup: Optional[bool]
pushover_on_intdown: Optional[bool]
pushover_on_intup: Optional[bool]
pushover_on_newdevice: Optional[bool]
pushover_on_pause: Optional[bool]
pushover_on_play: Optional[bool]
pushover_on_pmsupdate: Optional[bool]
pushover_on_resume: Optional[bool]
pushover_on_stop: Optional[bool]
pushover_on_watched: Optional[bool]
pushover_priority: Optional[bool]
pushover_sound: Optional[str]
```

```
class tautulli.models.settings.Scripts(**data)
```

```
    model_config: ClassVar[ConfigDict] = {}
```

```
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].
```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'scripts_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'scripts_folder':
FieldInfo(annotation=Union[str, NoneType], required=True), 'scripts_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True),
'scripts_on_buffer_script': FieldInfo(annotation=Union[str, NoneType],
required=True), 'scripts_on_concurrent': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'scripts_on_concurrent_script':
FieldInfo(annotation=Union[str, NoneType], required=True), 'scripts_on_created':
FieldInfo(annotation=Union[bool, NoneType], required=True),
'scripts_on_created_script': FieldInfo(annotation=Union[str, NoneType],
required=True), 'scripts_on_extdown': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'scripts_on_extdown_script': FieldInfo(annotation=Union[str,
NoneType], required=True), 'scripts_on_extup': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'scripts_on_extup_script':
FieldInfo(annotation=Union[str, NoneType], required=True), 'scripts_on_intdown':
FieldInfo(annotation=Union[bool, NoneType], required=True),
'scripts_on_intdown_script': FieldInfo(annotation=Union[str, NoneType],
required=True), 'scripts_on_intup': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'scripts_on_intup_script': FieldInfo(annotation=Union[str,
NoneType], required=True), 'scripts_on_newdevice': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'scripts_on_newdevice_script':
FieldInfo(annotation=Union[str, NoneType], required=True), 'scripts_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True),
'scripts_on_pause_script': FieldInfo(annotation=Union[str, NoneType],
required=True), 'scripts_on_play': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'scripts_on_play_script': FieldInfo(annotation=Union[str,
NoneType], required=True), 'scripts_on_pmsupdate': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'scripts_on_pmsupdate_script':
FieldInfo(annotation=Union[str, NoneType], required=True), 'scripts_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True),
'scripts_on_resume_script': FieldInfo(annotation=Union[str, NoneType],
required=True), 'scripts_on_stop': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'scripts_on_stop_script': FieldInfo(annotation=Union[str,
NoneType], required=True), 'scripts_on_watched': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'scripts_on_watched_script':
FieldInfo(annotation=Union[str, NoneType], required=True), 'scripts_timeout':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```

scripts_enabled: Optional[bool]

scripts_folder: Optional[str]

scripts_on_buffer: Optional[bool]

scripts_on_buffer_script: Optional[str]

scripts_on_concurrent: Optional[bool]

scripts_on_concurrent_script: Optional[str]

scripts_on_created: Optional[bool]

```

```
scripts_on_created_script: Optional[str]
scripts_on_extdown: Optional[bool]
scripts_on_extdown_script: Optional[str]
scripts_on_extup: Optional[bool]
scripts_on_extup_script: Optional[str]
scripts_on_intdown: Optional[bool]
scripts_on_intdown_script: Optional[str]
scripts_on_intup: Optional[bool]
scripts_on_intup_script: Optional[str]
scripts_on_newdevice: Optional[bool]
scripts_on_newdevice_script: Optional[str]
scripts_on_pause: Optional[bool]
scripts_on_pause_script: Optional[str]
scripts_on_play: Optional[bool]
scripts_on_play_script: Optional[str]
scripts_on_pmsupdate: Optional[bool]
scripts_on_pmsupdate_script: Optional[str]
scripts_on_resume: Optional[bool]
scripts_on_resume_script: Optional[str]
scripts_on_stop: Optional[bool]
scripts_on_stop_script: Optional[str]
scripts_on_watched: Optional[bool]
scripts_on_watched_script: Optional[str]
scripts_timeout: Optional[str]

class tautulli.models.settings.Settings(**data)
    Advanced: Optional[Advanced]
    Boxcar: Optional[Boxcar]
    Browser: Optional[Browser]
    Cloudinary: Optional[Cloudinary]
    Email: Optional[Email]
    Facebook: Optional[Facebook]
```

```
General: Optional[General]
Growl: Optional[Growl]
Hipchat: Optional[Hipchat]
IFTTT: Optional[IFTTT]
Join: Optional[Join]
Monitoring: Optional[Monitoring]
NMA: Optional[NMA]
Newsletter: Optional[Newsletter]
OSX_Notify: Optional[OSXNotify]
PMS: Optional[PMS]
Plex: Optional[Plex]
PlexWatch: Optional[PlexWatch]
Prowl: Optional[Prowl]
PushBullet: Optional[PushBullet]
Pushalot: Optional[Pushalot]
Pushover: Optional[Pushover]
Scripts: Optional[Scripts]
Slack: Optional[Slack]
Telegram: Optional[Telegram]
Twitter: Optional[Twitter]
XBMC: Optional[XBMC]
model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].
```

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'Advanced':
FieldInfo(annotation=Union[Advanced, NoneType], required=True), 'Boxcar':
FieldInfo(annotation=Union[Boxcar, NoneType], required=True), 'Browser':
FieldInfo(annotation=Union[Browser, NoneType], required=True), 'Clouldinary':
FieldInfo(annotation=Union[Clouldinary, NoneType], required=True), 'Email':
FieldInfo(annotation=Union[Email, NoneType], required=True), 'Facebook':
FieldInfo(annotation=Union[Facebook, NoneType], required=True), 'General':
FieldInfo(annotation=Union[General, NoneType], required=True), 'Growl':
FieldInfo(annotation=Union[Growl, NoneType], required=True), 'Hipchat':
FieldInfo(annotation=Union[Hipchat, NoneType], required=True), 'IFTTT':
FieldInfo(annotation=Union[IFTTT, NoneType], required=True), 'Join':
FieldInfo(annotation=Union[Join, NoneType], required=True), 'Monitoring':
FieldInfo(annotation=Union[Monitoring, NoneType], required=True), 'NMA':
FieldInfo(annotation=Union[NMA, NoneType], required=True), 'Newsletter':
FieldInfo(annotation=Union[Newsletter, NoneType], required=True), 'OSX_Notify':
FieldInfo(annotation=Union[OSXNotify, NoneType], required=True), 'PMS':
FieldInfo(annotation=Union[PMS, NoneType], required=True), 'Plex':
FieldInfo(annotation=Union[Plex, NoneType], required=True), 'PlexWatch':
FieldInfo(annotation=Union[PlexWatch, NoneType], required=True), 'Prowl':
FieldInfo(annotation=Union[Prowl, NoneType], required=True), 'PushBullet':
FieldInfo(annotation=Union[PushBullet, NoneType], required=True), 'Pushalot':
FieldInfo(annotation=Union[Pushalot, NoneType], required=True), 'Pushover':
FieldInfo(annotation=Union[Pushover, NoneType], required=True), 'Scripts':
FieldInfo(annotation=Union[Scripts, NoneType], required=True), 'Slack':
FieldInfo(annotation=Union[Slack, NoneType], required=True), 'Telegram':
FieldInfo(annotation=Union[Telegram, NoneType], required=True), 'Twitter':
FieldInfo(annotation=Union[Twitter, NoneType], required=True), 'XBMC':
FieldInfo(annotation=Union[XBMC, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
class tautulli.models.settings.Slack(**data)
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.


```

model_fields: ClassVar[dict[str, FieldInfo]] = {'slack_channel':
FieldInfo(annotation=Union[str, NoneType], required=True), 'slack_enabled':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_hook':
FieldInfo(annotation=Union[str, NoneType], required=True), 'slack_icon_emoji':
FieldInfo(annotation=Union[str, NoneType], required=True), 'slack_incl_pmslink':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_incl_poster':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_incl_subject':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_concurrent':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_created':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_extdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_extup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_intdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_intup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_newdevice':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_play':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'slack_username':
FieldInfo(annotation=Union[str, NoneType], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```

slack_channel: Optional[str]

slack_enabled: Optional[bool]

slack_hook: Optional[str]

slack_icon_emoji: Optional[str]

slack_incl_pmslink: Optional[bool]

slack_incl_poster: Optional[bool]

slack_incl_subject: Optional[bool]

slack_on_buffer: Optional[bool]

slack_on_concurrent: Optional[bool]

slack_on_created: Optional[bool]

slack_on_extdown: Optional[bool]

slack_on_extup: Optional[bool]

slack_on_intdown: Optional[bool]

slack_on_intup: Optional[bool]

slack_on_newdevice: Optional[bool]

```

```
slack_on_pause: Optional[bool]
slack_on_play: Optional[bool]
slack_on_pmsupdate: Optional[bool]
slack_on_resume: Optional[bool]
slack_on_stop: Optional[bool]
slack_on_watched: Optional[bool]
slack_username: Optional[str]
```

```
class tautulli.models.settings.Telegram(**data)
```

```
    model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
    model_fields: ClassVar[dict[str, FieldInfo]] = {'telegram_bot_token':
FieldInfo(annotation=Union[str, NoneType], required=True), 'telegram_chat_id':
FieldInfo(annotation=Union[str, NoneType], required=True),
'telegram_disable_web_preview': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'telegram_enabled': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'telegram_html_support': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'telegram_incl_poster': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'telegram_incl_subject':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'telegram_on_buffer':
FieldInfo(annotation=Union[bool, NoneType], required=True),
'telegram_on_concurrent': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'telegram_on_created': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'telegram_on_extdown': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'telegram_on_extup': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'telegram_on_intdown': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'telegram_on_intup': FieldInfo(annotation=Union[bool, NoneType],
required=True), 'telegram_on_newdevice': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'telegram_on_pause': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'telegram_on_play': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'telegram_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'telegram_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'telegram_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'telegram_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
telegram_bot_token: Optional[str]
telegram_chat_id: Optional[str]
telegram_disable_web_preview: Optional[bool]
telegram_enabled: Optional[bool]
```

```
telegram_html_support: Optional[bool]
telegram_incl_poster: Optional[bool]
telegram_incl_subject: Optional[bool]
telegram_on_buffer: Optional[bool]
telegram_on_concurrent: Optional[bool]
telegram_on_created: Optional[bool]
telegram_on_extdown: Optional[bool]
telegram_on_extup: Optional[bool]
telegram_on_intdown: Optional[bool]
telegram_on_intup: Optional[bool]
telegram_on_newdevice: Optional[bool]
telegram_on_pause: Optional[bool]
telegram_on_play: Optional[bool]
telegram_on_pmsupdate: Optional[bool]
telegram_on_resume: Optional[bool]
telegram_on_stop: Optional[bool]
telegram_on_watched: Optional[bool]

class tautulli.models.settings.Twitter(**data)
    model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].
```

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'twitter_access_token':
FieldInfo(annotation=Union[str, NoneType], required=True),
'twitter_access_token_secret': FieldInfo(annotation=Union[str, NoneType],
required=True), 'twitter_consumer_key': FieldInfo(annotation=Union[str, NoneType],
required=True), 'twitter_consumer_secret': FieldInfo(annotation=Union[str,
NoneType], required=True), 'twitter_enabled': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'twitter_incl_poster': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'twitter_incl_subject': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'twitter_on_buffer': FieldInfo(annotation=Union[bool,
NoneType], required=True), 'twitter_on_concurrent':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_created':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_extdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_extup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_intdown':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_intup':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_newdevice':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_pause':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_play':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_pmsupdate':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_resume':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_stop':
FieldInfo(annotation=Union[bool, NoneType], required=True), 'twitter_on_watched':
FieldInfo(annotation=Union[bool, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to [*FieldInfo*][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
twitter_access_token: Optional[str]

twitter_access_token_secret: Optional[str]

twitter_consumer_key: Optional[str]

twitter_consumer_secret: Optional[str]

twitter_enabled: Optional[bool]

twitter_incl_poster: Optional[bool]

twitter_incl_subject: Optional[bool]

twitter_on_buffer: Optional[bool]

twitter_on_concurrent: Optional[bool]

twitter_on_created: Optional[bool]

twitter_on_extdown: Optional[bool]

twitter_on_extup: Optional[bool]

twitter_on_intdown: Optional[bool]

twitter_on_intup: Optional[bool]

twitter_on_newdevice: Optional[bool]
```

```

twitter_on_pause: Optional[bool]

twitter_on_play: Optional[bool]

twitter_on_pmsupdate: Optional[bool]

twitter_on_resume: Optional[bool]

twitter_on_stop: Optional[bool]

twitter_on_watched: Optional[bool]

class tautulli.models.settings.XBMC(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'xbmc_enabled':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_host':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'xbmc_on_buffer':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_concurrent':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_created':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_extdown':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_extup':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_intdown':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_intup':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_newdevice':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_pause':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_play':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_pmsupdate':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_resume':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_stop':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_on_watched':
    FieldInfo(annotation=Union[bool, NoneType], required=True), 'xbmc_password':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'xbmc_username':
    FieldInfo(annotation=Union[str, NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

xbmc_enabled: Optional[bool]

xbmc_host: Optional[str]

xbmc_on_buffer: Optional[bool]

xbmc_on_concurrent: Optional[bool]

xbmc_on_created: Optional[bool]

xbmc_on_extdown: Optional[bool]

xbmc_on_extup: Optional[bool]

xbmc_on_intdown: Optional[bool]

```

```
xbmc_on_intup: Optional[bool]
xbmc_on_newdevice: Optional[bool]
xbmc_on_pause: Optional[bool]
xbmc_on_play: Optional[bool]
xbmc_on_pmsupdate: Optional[bool]
xbmc_on_resume: Optional[bool]
xbmc_on_stop: Optional[bool]
xbmc_on_watched: Optional[bool]
xbmc_password: Optional[str]
xbmc_username: Optional[str]
```

2.3.39 Stream Data

```
class tautulli.models.stream_data.StreamData(**data)
    aspect_ratio: Optional[str]
    audio_bitrate: Optional[int]
    audio_channels: Optional[int]
    audio_codec: Optional[str]
    audio_decision: Optional[str]
    bitrate: Optional[int]
    container: Optional[str]
    current_session: Optional[int]
    grandparent_title: Optional[str]
    media_type: Optional[str]
    model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].
```

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'aspect_ratio':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_bitrate':
FieldInfo(annotation=Union[int, NoneType], required=True), 'audio_channels':
FieldInfo(annotation=Union[int, NoneType], required=True), 'audio_codec':
FieldInfo(annotation=Union[str, NoneType], required=True), 'audio_decision':
FieldInfo(annotation=Union[str, NoneType], required=True), 'bitrate':
FieldInfo(annotation=Union[int, NoneType], required=True), 'container':
FieldInfo(annotation=Union[str, NoneType], required=True), 'current_session':
FieldInfo(annotation=Union[int, NoneType], required=True), 'grandparent_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'media_type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'optimized_version':
FieldInfo(annotation=Union[str, NoneType], required=True),
'optimized_version_profile': FieldInfo(annotation=Union[str, NoneType],
required=True), 'optimized_version_title': FieldInfo(annotation=Union[str,
NoneType], required=True), 'original_title': FieldInfo(annotation=Union[str,
NoneType], required=True), 'pre_tautulli': FieldInfo(annotation=Union[str,
NoneType], required=True), 'quality_profile': FieldInfo(annotation=Union[str,
NoneType], required=True), 'stream_audio_bitrate': FieldInfo(annotation=Union[int,
NoneType], required=True), 'stream_audio_channels': FieldInfo(annotation=Union[int,
NoneType], required=True), 'stream_audio_codec': FieldInfo(annotation=Union[str,
NoneType], required=True), 'stream_audio_decision': FieldInfo(annotation=Union[str,
NoneType], required=True), 'stream_bitrate': FieldInfo(annotation=Union[int,
NoneType], required=True), 'stream_container': FieldInfo(annotation=Union[str,
NoneType], required=True), 'stream_container_decision':
FieldInfo(annotation=Union[str, NoneType], required=True), 'stream_subtitle_codec':
FieldInfo(annotation=Union[str, NoneType], required=True),
'stream_subtitle_decision': FieldInfo(annotation=Union[str, NoneType],
required=True), 'stream_video_bitrate': FieldInfo(annotation=Union[int, NoneType],
required=True), 'stream_video_codec': FieldInfo(annotation=Union[str, NoneType],
required=True), 'stream_video_decision': FieldInfo(annotation=Union[str, NoneType],
required=True), 'stream_video_dynamic_range': FieldInfo(annotation=Union[str,
NoneType], required=True), 'stream_video_framerate':
FieldInfo(annotation=Union[str, NoneType], required=True),
'stream_video_full_resolution': FieldInfo(annotation=Union[str, NoneType],
required=True), 'stream_video_height': FieldInfo(annotation=Union[int, NoneType],
required=True), 'stream_video_width': FieldInfo(annotation=Union[int, NoneType],
required=True), 'subtitle_codec': FieldInfo(annotation=Union[str, NoneType],
required=True), 'subtitles': FieldInfo(annotation=Union[str, NoneType],
required=True), 'synced_version': FieldInfo(annotation=Union[str, NoneType],
required=True), 'synced_version_profile': FieldInfo(annotation=Union[str,
NoneType], required=True), 'title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'transcode_hw_decoding': FieldInfo(annotation=Union[str, NoneType],
required=True), 'transcode_hw_encoding': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_bitrate': FieldInfo(annotation=Union[int, NoneType],
required=True), 'video_codec': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_decision': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_dynamic_range': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_framerate': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_full_resolution': FieldInfo(annotation=Union[str, NoneType],
required=True), 'video_height': FieldInfo(annotation=Union[int, NoneType],
required=True), 'video_width': FieldInfo(annotation=Union[int, NoneType],
required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
optimized_version: Optional[str]
optimized_version_profile: Optional[str]
optimized_version_title: Optional[str]
original_title: Optional[str]
pre_tautulli: Optional[str]
quality_profile: Optional[str]
stream_audio_bitrate: Optional[int]
stream_audio_channels: Optional[int]
stream_audio_codec: Optional[str]
stream_audio_decision: Optional[str]
stream_bitrate: Optional[int]
stream_container: Optional[str]
stream_container_decision: Optional[str]
stream_subtitle_codec: Optional[str]
stream_subtitle_decision: Optional[str]
stream_video_bitrate: Optional[int]
stream_video_codec: Optional[str]
stream_video_decision: Optional[str]
stream_video_dynamic_range: Optional[str]
stream_video_framerate: Optional[str]
stream_video_full_resolution: Optional[str]
stream_video_height: Optional[int]
stream_video_width: Optional[int]
subtitle_codec: Optional[str]
subtitles: Optional[str]
synced_version: Optional[str]
synced_version_profile: Optional[str]
title: Optional[str]
transcode_hw_decoding: Optional[str]
transcode_hw_encoding: Optional[str]
```



```

video_bitrate: Optional[int]
video_codec: Optional[str]
video_decision: Optional[str]
video_dynamic_range: Optional[str]
video_framerate: Optional[str]
video_full_resolution: Optional[str]
video_height: Optional[int]
video_width: Optional[int]

```

2.3.40 Synced Items

```

class tautulli.models.synced_items.SyncedItems(**data)
    audio_bitrate: Optional[str]
    client_id: Optional[str]
    content_type: Optional[str]
    device_name: Optional[str]
    failure: Optional[str]
    item_complete_count: Optional[str]
    item_count: Optional[str]
    item_downloaded_count: Optional[str]
    item_downloaded_percent_complete: Optional[int]
    metadata_type: Optional[str]
    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

```

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'audio_bitrate':
FieldInfo(annotation=Union[str, NoneType], required=True), 'client_id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'content_type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'device_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'failure':
FieldInfo(annotation=Union[str, NoneType], required=True), 'item_complete_count':
FieldInfo(annotation=Union[str, NoneType], required=True), 'item_count':
FieldInfo(annotation=Union[str, NoneType], required=True), 'item_downloaded_count':
FieldInfo(annotation=Union[str, NoneType], required=True),
'item_downloaded_percent_complete': FieldInfo(annotation=Union[int, NoneType],
required=True), 'metadata_type': FieldInfo(annotation=Union[str, NoneType],
required=True), 'photo_quality': FieldInfo(annotation=Union[str, NoneType],
required=True), 'platform': FieldInfo(annotation=Union[str, NoneType],
required=True), 'rating_key': FieldInfo(annotation=Union[str, NoneType],
required=True), 'root_title': FieldInfo(annotation=Union[str, NoneType],
required=True), 'state': FieldInfo(annotation=Union[str, NoneType], required=True),
'sync_id': FieldInfo(annotation=Union[str, NoneType], required=True), 'sync_title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'total_size':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user_id':
FieldInfo(annotation=Union[str, NoneType], required=True), 'username':
FieldInfo(annotation=Union[str, NoneType], required=True), 'video_bitrate':
FieldInfo(annotation=Union[str, NoneType], required=True), 'video_quality':
FieldInfo(annotation=Union[str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
photo_quality: Optional[str]

platform: Optional[str]

rating_key: Optional[str]

root_title: Optional[str]

state: Optional[str]

sync_id: Optional[str]

sync_title: Optional[str]

total_size: Optional[str]

user: Optional[str]

user_id: Optional[str]

username: Optional[str]

video_bitrate: Optional[str]

video_quality: Optional[str]
```

2.3.41 Tautulli Info

```
class tautulli.models.tautulli_info.TautulliInfo(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'tautulli_branch':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'tautulli_commit':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'tautulli_install_type':
        FieldInfo(annotation=Union[str, NoneType], required=True), 'tautulli_platform':
        FieldInfo(annotation=Union[str, NoneType], required=True),
        'tautulli_platform_device_name': FieldInfo(annotation=Union[str, NoneType],
        required=True), 'tautulli_platform_linux_distro': FieldInfo(annotation=Union[str,
        NoneType], required=True), 'tautulli_platform_release':
        FieldInfo(annotation=Union[str, NoneType], required=True),
        'tautulli_platform_version': FieldInfo(annotation=Union[str, NoneType],
        required=True), 'tautulli_python_version': FieldInfo(annotation=Union[str,
        NoneType], required=True), 'tautulli_version': FieldInfo(annotation=Union[str,
        NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    tautulli_branch: Optional[str]

    tautulli_commit: Optional[str]

    tautulli_install_type: Optional[str]

    tautulli_platform: Optional[str]

    tautulli_platform_device_name: Optional[str]

    tautulli_platform_linux_distro: Optional[str]

    tautulli_platform_release: Optional[str]

    tautulli_platform_version: Optional[str]

    tautulli_python_version: Optional[str]

    tautulli_version: Optional[str]
```

2.3.42 Update Check

```
class tautulli.models.update_check.UpdateCheck(**data)

    install_type: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].
```

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'install_type':
FieldInfo(annotation=Union[str, NoneType], required=True), 'update':
FieldInfo(annotation=Union[bool, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
update: Optional[bool]
```

2.3.43 User

```
class tautulli.models.user.User(**data)
```

```
allow_guest: Optional[int]
```

```
deleted_user: Optional[int]
```

```
do_notify: Optional[int]
```

```
email: Optional[str]
```

```
friendly_name: Optional[str]
```

```
is_active: Optional[int]
```

```
is_admin: Optional[str]
```

```
is_allow_sync: Optional[int]
```

```
is_home_user: Optional[int]
```

```
is_restricted: Optional[int]
```

```
keep_history: Optional[int]
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'allow_guest':
FieldInfo(annotation=Union[int, NoneType], required=True), 'deleted_user':
FieldInfo(annotation=Union[int, NoneType], required=True), 'do_notify':
FieldInfo(annotation=Union[int, NoneType], required=True), 'email':
FieldInfo(annotation=Union[str, NoneType], required=True), 'friendly_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'is_active':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_admin':
FieldInfo(annotation=Union[str, NoneType], required=True), 'is_allow_sync':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_home_user':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_restricted':
FieldInfo(annotation=Union[int, NoneType], required=True), 'keep_history':
FieldInfo(annotation=Union[int, NoneType], required=True), 'row_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'shared_libraries':
FieldInfo(annotation=Union[List, NoneType], required=True), 'user_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'user_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'username':
FieldInfo(annotation=Union[str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to [*FieldInfo*][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
row_id: Optional[int]

shared_libraries: Optional[List]

user_id: Optional[int]

user_thumb: Optional[str]

username: Optional[str]
```

2.3.44 Users

```
class tautulli.models.users.User(**data)
```

```
    allow_guest: Optional[int]

    do_notify: Optional[int]

    email: Optional[str]

    filter_all: Optional[str]

    filter_movies: Optional[str]

    filter_music: Optional[str]

    filter_photos: Optional[str]

    filter_tv: Optional[str]

    friendly_name: Optional[str]

    is_active: Optional[int]

    is_admin: Optional[int]

    is_allow_sync: Optional[int]

    is_home_user: Optional[int]

    is_restricted: Optional[int]

    keep_history: Optional[int]

    model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to [*ConfigDict*][pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'allow_guest':
FieldInfo(annotation=Union[int, NoneType], required=True), 'do_notify':
FieldInfo(annotation=Union[int, NoneType], required=True), 'email':
FieldInfo(annotation=Union[str, NoneType], required=True), 'filter_all':
FieldInfo(annotation=Union[str, NoneType], required=True), 'filter_movies':
FieldInfo(annotation=Union[str, NoneType], required=True), 'filter_music':
FieldInfo(annotation=Union[str, NoneType], required=True), 'filter_photos':
FieldInfo(annotation=Union[str, NoneType], required=True), 'filter_tv':
FieldInfo(annotation=Union[str, NoneType], required=True), 'friendly_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'is_active':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_admin':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_allow_sync':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_home_user':
FieldInfo(annotation=Union[int, NoneType], required=True), 'is_restricted':
FieldInfo(annotation=Union[int, NoneType], required=True), 'keep_history':
FieldInfo(annotation=Union[int, NoneType], required=True), 'row_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'shared_libraries':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'username':
FieldInfo(annotation=Union[str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
row_id: Optional[int]
```

```
shared_libraries: Optional[List[str]]
```

```
thumb: Optional[str]
```

```
user_id: Optional[int]
```

```
username: Optional[str]
```

2.3.45 Users Table

```
class tautulli.models.users_table.Datum(**data)
```

```
allow_guest: Optional[int]
```

```
do_notify: Optional[int]
```

```
duration: Optional[int]
```

```
email: Optional[str]
```

```
friendly_name: Optional[str]
```

```
guid: Optional[str]
```

```
history_row_id: Optional[int]
```

```
ip_address: Optional[str]
```

```

is_active: Optional[int]

keep_history: Optional[int]

last_played: Optional[str]

last_seen: Optional[int]

live: Optional[int]

media_index: Optional[Union[int, str]]

media_type: Optional[str]

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'allow_guest':
FieldInfo(annotation=Union[int, NoneType], required=True), 'do_notify':
FieldInfo(annotation=Union[int, NoneType], required=True), 'duration':
FieldInfo(annotation=Union[int, NoneType], required=True), 'email':
FieldInfo(annotation=Union[str, NoneType], required=True), 'friendly_name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'guid':
FieldInfo(annotation=Union[str, NoneType], required=True), 'history_row_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'ip_address':
FieldInfo(annotation=Union[str, NoneType], required=True), 'is_active':
FieldInfo(annotation=Union[int, NoneType], required=True), 'keep_history':
FieldInfo(annotation=Union[int, NoneType], required=True), 'last_played':
FieldInfo(annotation=Union[str, NoneType], required=True), 'last_seen':
FieldInfo(annotation=Union[int, NoneType], required=True), 'live':
FieldInfo(annotation=Union[int, NoneType], required=True), 'media_index':
FieldInfo(annotation=Union[int, str, NoneType], required=True), 'media_type':
FieldInfo(annotation=Union[str, NoneType], required=True),
'originally_available_at': FieldInfo(annotation=Union[str, NoneType],
required=True), 'parent_media_index': FieldInfo(annotation=Union[int, str,
NoneType], required=True), 'parent_title': FieldInfo(annotation=Union[str,
NoneType], required=True), 'platform': FieldInfo(annotation=Union[str, NoneType],
required=True), 'player': FieldInfo(annotation=Union[str, NoneType],
required=True), 'plays': FieldInfo(annotation=Union[int, NoneType], required=True),
'rating_key': FieldInfo(annotation=Union[int, NoneType], required=True), 'row_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'title':
FieldInfo(annotation=Union[str, NoneType], required=True), 'transcode_decision':
FieldInfo(annotation=Union[str, NoneType], required=True), 'user_id':
FieldInfo(annotation=Union[int, NoneType], required=True), 'user_thumb':
FieldInfo(annotation=Union[str, NoneType], required=True), 'username':
FieldInfo(annotation=Union[str, NoneType], required=True), 'year':
FieldInfo(annotation=Union[Any, NoneType], required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

originally_available_at: Optional[str]

```

```
parent_media_index: Optional[Union[int, str]]
parent_title: Optional[str]
platform: Optional[str]
player: Optional[str]
plays: Optional[int]
rating_key: Optional[int]
row_id: Optional[int]
thumb: Optional[str]
title: Optional[str]
transcode_decision: Optional[str]
user_id: Optional[int]
user_thumb: Optional[str]
username: Optional[str]
year: Optional[Any]
```

```
class tautulli.models.users_table.UsersTable(**data)
```

```
data: Optional[List[Datum]]
```

```
draw: Optional[int]
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'data':
FieldInfo(annotation=Union[List[tautulli.models.users_table.Datum], NoneType],
required=True), 'draw': FieldInfo(annotation=Union[int, NoneType], required=True),
'recordsFiltered': FieldInfo(annotation=Union[int, NoneType], required=True),
'recordsTotal': FieldInfo(annotation=Union[int, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
recordsFiltered: Optional[int]
```

```
recordsTotal: Optional[int]
```


2.3.46 User IPs

```
class tautulli.models.user_ips.Datum(**data)

    first_seen: Optional[int]

    friendly_name: Optional[str]

    guid: Optional[str]

    id: Optional[int]

    ip_address: Optional[str]

    last_played: Optional[str]

    last_seen: Optional[int]

    live: Optional[int]

    media_index: Optional[int]

    media_type: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'first_seen':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'friendly_name':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'guid':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'id':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'ip_address':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'last_played':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'last_seen':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'live':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'media_index':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'media_type':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'media_type':
    FieldInfo(annotation=Union[str, NoneType], required=True),
    'originally_available_at': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'parent_media_index': FieldInfo(annotation=Union[int, NoneType],
    required=True), 'parent_title': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'platform': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'play_count': FieldInfo(annotation=Union[int, NoneType],
    required=True), 'player': FieldInfo(annotation=Union[str, NoneType],
    required=True), 'rating_key': FieldInfo(annotation=Union[int, NoneType],
    required=True), 'thumb': FieldInfo(annotation=Union[str, NoneType], required=True),
    'transcode_decision': FieldInfo(annotation=Union[str, NoneType], required=True),
    'user_id': FieldInfo(annotation=Union[int, NoneType], required=True), 'year':
    FieldInfo(annotation=Union[int, NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    originally_available_at: Optional[str]
```

```
parent_media_index: Optional[int]
parent_title: Optional[str]
platform: Optional[str]
play_count: Optional[int]
player: Optional[str]
rating_key: Optional[int]
thumb: Optional[str]
transcode_decision: Optional[str]
user_id: Optional[int]
year: Optional[int]

class tautulli.models.user_ips.UserIPs(**data)

    data: Optional[List[Datum]]
    draw: Optional[int]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'data':
        FieldInfo(annotation=Union[List[tautulli.models.user_ips.Datum], NoneType],
        required=True), 'draw': FieldInfo(annotation=Union[int, NoneType], required=True),
        'recordsFiltered': FieldInfo(annotation=Union[int, NoneType], required=True),
        'recordsTotal': FieldInfo(annotation=Union[int, NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    recordsFiltered: Optional[int]
    recordsTotal: Optional[int]
```

2.3.47 User Logins

```
class tautulli.models.user_logins.Datum(**data)

    browser: Optional[str]
    friendly_name: Optional[str]
    host: Optional[str]
    ip_address: Optional[str]
```

```

model_config: ClassVar[ConfigDict] = {}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'browser':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'friendly_name':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'host':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'ip_address':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'os':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'timestamp':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'user':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'user_agent':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'user_group':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'user_id':
    FieldInfo(annotation=Union[int, NoneType], required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

os: Optional[str]

timestamp: Optional[int]

user: Optional[str]

user_agent: Optional[str]

user_group: Optional[str]

user_id: Optional[int]

class tautulli.models.user_logins.UserLogins(**data)

    data: Optional[List[Datum]]

    draw: Optional[int]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'data':
        FieldInfo(annotation=Union[List[tautulli.models.user_logins.Datum], NoneType],
        required=True), 'draw': FieldInfo(annotation=Union[int, NoneType], required=True),
        'recordsFiltered': FieldInfo(annotation=Union[int, NoneType], required=True),
        'recordsTotal': FieldInfo(annotation=Union[int, NoneType], required=True)}

        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    recordsFiltered: Optional[int]

    recordsTotal: Optional[int]

```

2.3.48 Usernames

```
class tautulli.models.usernames.UserName(**data)

    friendly_name: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'friendly_name':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'user_id':
    FieldInfo(annotation=Union[int, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    user_id: Optional[int]
```

2.3.49 User Player Stats

```
class tautulli.models.user_player_stats.UserPlayerStats(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [Config-
        Dict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'platform':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'platform_name':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'player_name':
    FieldInfo(annotation=Union[str, NoneType], required=True), 'result_id':
    FieldInfo(annotation=Union[int, NoneType], required=True), 'total_plays':
    FieldInfo(annotation=Union[int, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [Field-
        Info][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    platform: Optional[str]

    platform_name: Optional[str]

    player_name: Optional[str]

    result_id: Optional[int]

    total_plays: Optional[int]
```

2.3.50 User Watch Time Stats

```
class tautulli.models.user_watch_time_stats.UserWatchTimeStats(**data)

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

    model_fields: ClassVar[dict[str, FieldInfo]] = {'query_days':
        FieldInfo(annotation=Union[int, NoneType], required=True), 'total_plays':
        FieldInfo(annotation=Union[int, NoneType], required=True), 'total_time':
        FieldInfo(annotation=Union[int, NoneType], required=True)}
        Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

        This replaces Model.__fields__ from Pydantic V1.

    query_days: Optional[int]

    total_plays: Optional[int]

    total_time: Optional[int]
```

2.3.51 WHOIS Lookup

```
class tautulli.models.whois_lookup.Net(**data)

    address: Optional[str]

    cidr: Optional[str]

    city: Optional[str]

    country: Optional[str]

    created: Optional[str]

    description: Optional[str]

    emails: Optional[List[str]]

    handle: Optional[str]

    model_config: ClassVar[ConfigDict] = {}
        Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].
```

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'address':
FieldInfo(annotation=Union[str, NoneType], required=True), 'cidr':
FieldInfo(annotation=Union[str, NoneType], required=True), 'city':
FieldInfo(annotation=Union[str, NoneType], required=True), 'country':
FieldInfo(annotation=Union[str, NoneType], required=True), 'created':
FieldInfo(annotation=Union[str, NoneType], required=True), 'description':
FieldInfo(annotation=Union[str, NoneType], required=True), 'emails':
FieldInfo(annotation=Union[List[str], NoneType], required=True), 'handle':
FieldInfo(annotation=Union[str, NoneType], required=True), 'name':
FieldInfo(annotation=Union[str, NoneType], required=True), 'postal_code':
FieldInfo(annotation=Union[str, NoneType], required=True), 'range':
FieldInfo(annotation=Union[str, NoneType], required=True), 'state':
FieldInfo(annotation=Union[str, NoneType], required=True), 'updated':
FieldInfo(annotation=Union[str, NoneType], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
name: Optional[str]

postal_code: Optional[str]

range: Optional[str]

state: Optional[str]

updated: Optional[str]
```

```
class tautulli.models.whois_lookup.WHOISLookup(**data)
```

```
host: Optional[str]
```

```
model_config: ClassVar[ConfigDict] = {}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict]*[pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'host':
FieldInfo(annotation=Union[str, NoneType], required=True), 'nets':
FieldInfo(annotation=Union[List[tautulli.models.whois_lookup.Net], NoneType],
required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo]*[pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
nets: Optional[List[Net]]
```

2.4 Tools

class tautulli.tools.api_helper.**APIShortcuts**(*api*)

get_library_by_name(*library_name*)

Get a Plex Media Server library using its name

Returns

Dict of data

Return type

dict

ping()

Ping the Tautulli server

Returns

True if successful, *False* if unsuccessful

Return type

bool

ping_plex(*pms_url=None*)

Ping the Plex Media Server :type pms_url: Optional[str]:param pms_url: URL of the Plex Media Server, defaults to default PMS URL :type pms_url: str, optional :returns: *True* if successful, *False* if unsuccessful :rtype: bool

property activity_summary: dict

Get a summary of current activity on the Plex Media Server

Returns

Dict of data

Return type

dict

property activity_summary_message: str

Get a summary message of current activity on the Plex Media Server

Returns

Activity summary message

Return type

str

property api_version: str

Get the Tautulli API version

Returns

API version

Return type

str

property has_plex_pass: bool

Check if the Plex Media Server has Plex Pass

Returns

True if successful, *False* if unsuccessful

Return type
bool

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

t

- `tautulli.models.activity`, 62
- `tautulli.models.collections_table`, 72
- `tautulli.models.date_formats`, 74
- `tautulli.models.docs`, 74
- `tautulli.models.export_fields`, 79
- `tautulli.models.geo_ip_lookup`, 80
- `tautulli.models.history`, 81
- `tautulli.models.home_stats`, 84
- `tautulli.models.libraries`, 87
- `tautulli.models.libraries_table`, 88
- `tautulli.models.library`, 86
- `tautulli.models.library_media_info`, 90
- `tautulli.models.library_names`, 92
- `tautulli.models.library_user_stats`, 93
- `tautulli.models.library_watch_time_stats`, 93
- `tautulli.models.logs`, 94
- `tautulli.models.metadata`, 94
- `tautulli.models.new_rating_keys`, 102
- `tautulli.models.newsletter`, 110
- `tautulli.models.newsletter_config`, 103
- `tautulli.models.newsletter_log`, 108
- `tautulli.models.notification_log`, 110
- `tautulli.models.notifier_parameters`, 112
- `tautulli.models.notifiers`, 112
- `tautulli.models.old_rating_keys`, 113
- `tautulli.models.playlists_table`, 113
- `tautulli.models.plex_log`, 115
- `tautulli.models.pms_update`, 115
- `tautulli.models.recently_added`, 116
- `tautulli.models.registered_device`, 116
- `tautulli.models.search_results`, 118
- `tautulli.models.server_id`, 134
- `tautulli.models.server_identity`, 135
- `tautulli.models.server_info`, 135
- `tautulli.models.server_list`, 136
- `tautulli.models.server_status`, 137
- `tautulli.models.servers_info`, 137
- `tautulli.models.settings`, 138
- `tautulli.models.stream_data`, 178
- `tautulli.models.synced_items`, 181
- `tautulli.models.tautulli_info`, 183
- `tautulli.models.update_check`, 183
- `tautulli.models.user`, 184
- `tautulli.models.user_ips`, 189
- `tautulli.models.user_logins`, 190
- `tautulli.models.user_player_stats`, 192
- `tautulli.models.user_watch_time_stats`, 193
- `tautulli.models.usernames`, 192
- `tautulli.models.users`, 185
- `tautulli.models.users_table`, 186
- `tautulli.models.whois_lookup`, 193

A

- accuracy (*tautulli.models.geo_ip_lookup.GeoIPLookup* attribute), 80
- active (*tautulli.models.newsletter.Newsletter* attribute), 110
- active (*tautulli.models.newsletter_config.NewsletterConfig* attribute), 106
- active (*tautulli.models.notifiers.Notifier* attribute), 112
- Activity (class in *tautulli.models.activity*), 62
- activity() (*tautulli.api.json_api.RawAPI* method), 3
- activity() (*tautulli.api.object_api.ObjectAPI* method), 33
- activity_summary (*tautulli.tools.api_helper.APIShortcuts* property), 195
- activity_summary_message (*tautulli.tools.api_helper.APIShortcuts* property), 195
- ActivitySummary (class in *tautulli.models.activity*), 62
- actors (*tautulli.models.activity.Session* attribute), 63
- actors (*tautulli.models.metadata.Metadata* attribute), 96
- actors (*tautulli.models.search_results.EpisodeItem* attribute), 118
- actors (*tautulli.models.search_results.MovieItem* attribute), 122
- actors (*tautulli.models.search_results.SeasonItem* attribute), 127
- actors (*tautulli.models.search_results.ShowItem* attribute), 129
- add_live_tv_library (*tautulli.models.settings.Advanced* attribute), 138
- add_newsletter_config (*tautulli.models.docs.Docs* attribute), 74
- add_newsletter_config() (*tautulli.api.json_api.RawAPI* method), 3
- add_newsletter_config() (*tautulli.api.object_api.ObjectAPI* method), 33
- add_notifier_config (*tautulli.models.docs.Docs* attribute), 74
- add_notifier_config() (*tautulli.api.json_api.RawAPI* method), 3
- add_notifier_config() (*tautulli.api.object_api.ObjectAPI* method), 33
- added_at (*tautulli.models.activity.Session* attribute), 63
- added_at (*tautulli.models.library_media_info.Datum* attribute), 90
- added_at (*tautulli.models.metadata.Metadata* attribute), 96
- added_at (*tautulli.models.search_results.EpisodeItem* attribute), 118
- added_at (*tautulli.models.search_results.MovieItem* attribute), 122
- added_at (*tautulli.models.search_results.SeasonItem* attribute), 127
- added_at (*tautulli.models.search_results.ShowItem* attribute), 129
- addedAt (*tautulli.models.collections_table.Datum* attribute), 72
- addedAt (*tautulli.models.playlists_table.Datum* attribute), 113
- address (*tautulli.models.whois_lookup.Net* attribute), 193
- Advanced (class in *tautulli.models.settings*), 138
- Advanced (*tautulli.models.settings.Settings* attribute), 170
- agent (*tautulli.models.libraries.LibrariesEntry* attribute), 87
- agent (*tautulli.models.library_names.LibraryName* attribute), 92
- agent_id (*tautulli.models.newsletter.Newsletter* attribute), 110
- agent_id (*tautulli.models.newsletter_config.NewsletterConfig* attribute), 106
- agent_id (*tautulli.models.newsletter_log.Datum* attribute), 108
- agent_id (*tautulli.models.notification_log.Datum* attribute), 110
- agent_id (*tautulli.models.notifiers.Notifier* attribute), 112
- agent_label (*tautulli.models.newsletter.Newsletter* attribute), 110

tribute), 110
 agent_label (tautulli.models.newsletter_config.NewsletterConfig attribute), 106
 agent_label (tautulli.models.notifiers.Notifier attribute), 112
 agent_name (tautulli.models.newsletter.Newsletter attribute), 110
 agent_name (tautulli.models.newsletter_config.NewsletterConfig attribute), 106
 agent_name (tautulli.models.newsletter_log.Datum attribute), 108
 agent_name (tautulli.models.notification_log.Datum attribute), 110
 agent_name (tautulli.models.notifiers.Notifier attribute), 112
 album (tautulli.models.search_results.ResultsList attribute), 126
 allow_guest (tautulli.models.activity.Session attribute), 63
 allow_guest (tautulli.models.user.User attribute), 184
 allow_guest (tautulli.models.users.User attribute), 185
 allow_guest (tautulli.models.users_table.Datum attribute), 186
 allow_guest_access (tautulli.models.settings.General attribute), 144
 anon_redirect (tautulli.models.settings.General attribute), 145
 api_enabled (tautulli.models.settings.General attribute), 145
 api_key (tautulli.models.settings.General attribute), 145
 api_sql (tautulli.models.settings.General attribute), 145
 api_version (tautulli.tools.api_helper.APIShortcuts property), 195
 APIShortcuts (class in tautulli.tools.api_helper), 195
 arnold (tautulli.api.json_api.RawAPI property), 30
 arnold (tautulli.api.object_api.ObjectAPI property), 59
 arnold (tautulli.models.docs.Docs attribute), 74
 art (tautulli.models.activity.Session attribute), 63
 art (tautulli.models.collections_table.Datum attribute), 73
 art (tautulli.models.home_stats.Row attribute), 84
 art (tautulli.models.libraries.LibrariesEntry attribute), 87
 art (tautulli.models.metadata.Metadata attribute), 96
 art (tautulli.models.search_results.EpisodeItem attribute), 118
 art (tautulli.models.search_results.MovieItem attribute), 122
 art (tautulli.models.search_results.SeasonItem attribute), 127
 art (tautulli.models.search_results.ShowItem attribute), 129
 artist (tautulli.models.search_results.ResultsList attribute), 126
 aspect_ratio (tautulli.models.activity.Session attribute), 63
 aspect_ratio (tautulli.models.metadata.MediaInfoItem attribute), 95
 aspect_ratio (tautulli.models.search_results.MediaInfoItem attribute), 121
 aspect_ratio (tautulli.models.stream_data.StreamData attribute), 178
 audience_rating (tautulli.models.activity.Session attribute), 63
 audience_rating (tautulli.models.metadata.Metadata attribute), 96
 audience_rating (tautulli.models.search_results.EpisodeItem attribute), 118
 audience_rating (tautulli.models.search_results.MovieItem attribute), 122
 audience_rating (tautulli.models.search_results.SeasonItem attribute), 127
 audience_rating (tautulli.models.search_results.ShowItem attribute), 129
 audience_rating_image (tautulli.models.activity.Session attribute), 63
 audience_rating_image (tautulli.models.metadata.Metadata attribute), 96
 audience_rating_image (tautulli.models.search_results.EpisodeItem attribute), 118
 audience_rating_image (tautulli.models.search_results.MovieItem attribute), 123
 audience_rating_image (tautulli.models.search_results.SeasonItem attribute), 127
 audience_rating_image (tautulli.models.search_results.ShowItem attribute), 129
 audio_bitrate (tautulli.models.activity.Session attribute), 63
 audio_bitrate (tautulli.models.metadata.Stream attribute), 100
 audio_bitrate (tautulli.models.search_results.Stream attribute), 132
 audio_bitrate (tautulli.models.stream_data.StreamData attribute), 178
 audio_bitrate (tautulli.models.synced_items.SyncedItems attribute), 181
 audio_bitrate_mode (tautulli.models.activity.Session attribute), 63
 audio_bitrate_mode (tautulli.models.metadata.Stream

- attribute), 100
- audio_bitrate_mode (tautulli.models.search_results.Stream attribute), 132
- audio_channel_layout (tautulli.models.activity.Session attribute), 63
- audio_channel_layout (tautulli.models.metadata.MediaInfoItem attribute), 95
- audio_channel_layout (tautulli.models.metadata.Stream attribute), 100
- audio_channel_layout (tautulli.models.search_results.MediaInfoItem attribute), 121
- audio_channel_layout (tautulli.models.search_results.Stream attribute), 132
- audio_channels (tautulli.models.activity.Session attribute), 63
- audio_channels (tautulli.models.library_media_info.Datum attribute), 90
- audio_channels (tautulli.models.metadata.MediaInfoItem attribute), 95
- audio_channels (tautulli.models.metadata.Stream attribute), 100
- audio_channels (tautulli.models.search_results.MediaInfoItem attribute), 121
- audio_channels (tautulli.models.search_results.Stream attribute), 132
- audio_channels (tautulli.models.stream_data.StreamData attribute), 178
- audio_codec (tautulli.models.activity.Session attribute), 63
- audio_codec (tautulli.models.library_media_info.Datum attribute), 90
- audio_codec (tautulli.models.metadata.MediaInfoItem attribute), 95
- audio_codec (tautulli.models.metadata.Stream attribute), 100
- audio_codec (tautulli.models.search_results.MediaInfoItem attribute), 121
- audio_codec (tautulli.models.search_results.Stream attribute), 132
- audio_codec (tautulli.models.stream_data.StreamData attribute), 178
- audio_decision (tautulli.models.activity.Session attribute), 63
- audio_decision (tautulli.models.stream_data.StreamData attribute), 178
- audio_language (tautulli.models.activity.Session attribute), 63
- audio_language (tautulli.models.metadata.Stream attribute), 100
- audio_language (tautulli.models.search_results.Stream attribute), 132
- audio_language_code (tautulli.models.activity.Session attribute), 64
- audio_language_code (tautulli.models.metadata.Stream attribute), 100
- audio_language_code (tautulli.models.search_results.Stream attribute), 132
- audio_profile (tautulli.models.activity.Session attribute), 64
- audio_profile (tautulli.models.metadata.MediaInfoItem attribute), 95
- audio_profile (tautulli.models.metadata.Stream attribute), 100
- audio_profile (tautulli.models.search_results.MediaInfoItem attribute), 121
- audio_profile (tautulli.models.search_results.Stream attribute), 132
- audio_sample_rate (tautulli.models.activity.Session attribute), 64
- audio_sample_rate (tautulli.models.metadata.Stream attribute), 100
- audio_sample_rate (tautulli.models.search_results.Stream attribute), 132
- ## B
- backup_config (tautulli.models.docs.Docs attribute), 74
- backup_config() (tautulli.api.json_api.RawAPI method), 3
- backup_config() (tautulli.api.object_api.ObjectAPI method), 33
- backup_database() (tautulli.api.json_api.RawAPI method), 4
- backup_database() (tautulli.api.object_api.ObjectAPI method), 34
- backup_days (tautulli.models.settings.General attribute), 145
- backup_db (tautulli.models.docs.Docs attribute), 74
- backup_dir (tautulli.models.settings.General attribute), 145
- backup_interval (tautulli.models.settings.General attribute), 145
- bandwidth (tautulli.models.activity.Session attribute), 64
- banner (tautulli.models.activity.Session attribute), 64

- banner (*tautulli.models.metadata.Metadata* attribute), 96
- banner (*tautulli.models.search_results.EpisodeItem* attribute), 118
- banner (*tautulli.models.search_results.MovieItem* attribute), 123
- banner (*tautulli.models.search_results.SeasonItem* attribute), 127
- banner (*tautulli.models.search_results.ShowItem* attribute), 129
- bcc (*tautulli.models.newsletter_config.EmailConfig* attribute), 104
- bif_thumb (*tautulli.models.activity.Session* attribute), 64
- bitrate (*tautulli.models.activity.Session* attribute), 64
- bitrate (*tautulli.models.library_media_info.Datum* attribute), 90
- bitrate (*tautulli.models.metadata.MediaInfoItem* attribute), 95
- bitrate (*tautulli.models.search_results.MediaInfoItem* attribute), 121
- bitrate (*tautulli.models.stream_data.StreamData* attribute), 178
- body (*tautulli.models.newsletter_config.NewsletterConfig* attribute), 106
- body_text (*tautulli.models.newsletter_log.Datum* attribute), 108
- body_text (*tautulli.models.notification_log.Datum* attribute), 110
- Boxcar (class in *tautulli.models.settings*), 139
- Boxcar (*tautulli.models.settings.Settings* attribute), 170
- boxcar_enabled (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_buffer (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_concurrent (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_created (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_extdown (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_extup (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_intdown (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_intup (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_newdevice (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_pause (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_play (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_pmsupdate (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_resume (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_stop (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_on_watched (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_sound (*tautulli.models.settings.Boxcar* attribute), 139
- boxcar_token (*tautulli.models.settings.Boxcar* attribute), 139
- Browser (class in *tautulli.models.settings*), 140
- Browser (*tautulli.models.settings.Settings* attribute), 170
- browser (*tautulli.models.user_logins.Datum* attribute), 190
- browser_auto_hide_delay (*tautulli.models.settings.Browser* attribute), 140
- browser_enabled (*tautulli.models.settings.Browser* attribute), 140
- browser_on_buffer (*tautulli.models.settings.Browser* attribute), 140
- browser_on_concurrent (*tautulli.models.settings.Browser* attribute), 140
- browser_on_created (*tautulli.models.settings.Browser* attribute), 140
- browser_on_extdown (*tautulli.models.settings.Browser* attribute), 140
- browser_on_extup (*tautulli.models.settings.Browser* attribute), 140
- browser_on_intdown (*tautulli.models.settings.Browser* attribute), 140
- browser_on_intup (*tautulli.models.settings.Browser* attribute), 140
- browser_on_newdevice (*tautulli.models.settings.Browser* attribute), 140
- browser_on_pause (*tautulli.models.settings.Browser* attribute), 140
- browser_on_play (*tautulli.models.settings.Browser* attribute), 140
- browser_on_pmsupdate (*tautulli.models.settings.Browser* attribute), 140
- browser_on_resume (*tautulli.models.settings.Browser* attribute), 140
- browser_on_stop (*tautulli.models.settings.Browser* attribute), 140
- browser_on_watched (*tautulli.models.settings.Browser* attribute), 140
- buffer_threshold (*tautulli.models.settings.Monitoring* attribute), 154
- buffer_wait (*tautulli.models.settings.Monitoring* attribute), 154
- build_summary_from_activity_json() (in module *tautulli.models.activity*), 72
- build_summary_from_activity_object() (in module *tautulli.models.activity*), 72

C

- `cache_dir` (*tautulli.models.settings.General* attribute), 145
- `cache_images` (*tautulli.models.settings.General* attribute), 145
- `cache_size_mb` (*tautulli.models.settings.Advanced* attribute), 138
- `cc` (*tautulli.models.newsletter_config.EmailConfig* attribute), 104
- `changelog_added` (*tautulli.models.pms_update.PMSUpdate* attribute), 115
- `changelog_fixed` (*tautulli.models.pms_update.PMSUpdate* attribute), 115
- `channel_call_sign` (*tautulli.models.activity.Session* attribute), 64
- `channel_call_sign` (*tautulli.models.metadata.MediaInfoItem* attribute), 95
- `channel_call_sign` (*tautulli.models.search_results.MediaInfoItem* attribute), 121
- `channel_identifier` (*tautulli.models.activity.Session* attribute), 64
- `channel_identifier` (*tautulli.models.metadata.MediaInfoItem* attribute), 95
- `channel_identifier` (*tautulli.models.search_results.MediaInfoItem* attribute), 121
- `channel_stream` (*tautulli.models.activity.Session* attribute), 64
- `channel_thumb` (*tautulli.models.activity.Session* attribute), 64
- `channel_thumb` (*tautulli.models.metadata.MediaInfoItem* attribute), 95
- `channel_thumb` (*tautulli.models.search_results.MediaInfoItem* attribute), 121
- `check_github` (*tautulli.models.settings.General* attribute), 145
- `check_github_cache_seconds` (*tautulli.models.settings.Advanced* attribute), 138
- `check_github_interval` (*tautulli.models.settings.General* attribute), 145
- `check_github_on_startup` (*tautulli.models.settings.General* attribute), 145
- `child_count` (*tautulli.models.libraries.LibrariesEntry* attribute), 87
- `child_count` (*tautulli.models.libraries_table.Datum* attribute), 88
- `child_count` (*tautulli.models.library.Library* attribute), 86
- `childCount` (*tautulli.models.collections_table.Datum* attribute), 73
- `children_count` (*tautulli.models.activity.Session* attribute), 64
- `children_count` (*tautulli.models.metadata.Metadata* attribute), 96
- `children_count` (*tautulli.models.search_results.EpisodeItem* attribute), 118
- `children_count` (*tautulli.models.search_results.MovieItem* attribute), 123
- `children_count` (*tautulli.models.search_results.SeasonItem* attribute), 127
- `children_count` (*tautulli.models.search_results.ShowItem* attribute), 129
- `cidr` (*tautulli.models.whois_lookup.Net* attribute), 193
- `city` (*tautulli.models.geo_ip_lookup.GeoIPLookup* attribute), 80
- `city` (*tautulli.models.whois_lookup.Net* attribute), 193
- `cleanup_files` (*tautulli.models.settings.General* attribute), 145
- `client_id` (*tautulli.models.synced_items.SyncedItems* attribute), 181
- `clientIdentifier` (*tautulli.models.server_list.ServerListEntry* attribute), 136
- `Cloudinary` (class in *tautulli.models.settings*), 141
- `Cloudinary` (*tautulli.models.settings.Settings* attribute), 170
- `cloudinary_api_key` (*tautulli.models.settings.Cloudinary* attribute), 141
- `cloudinary_api_secret` (*tautulli.models.settings.Cloudinary* attribute), 141
- `cloudinary_cloud_name` (*tautulli.models.settings.Cloudinary* attribute), 141
- `code` (*tautulli.models.geo_ip_lookup.GeoIPLookup* attribute), 80
- `collection` (*tautulli.models.search_results.ResultsList* attribute), 126
- `collectionMode` (*tautulli.models.collections_table.Datum* attribute), 73
- `collections` (*tautulli.models.activity.Session* attribute), 64
- `collections` (*tautulli.models.metadata.Metadata* attribute), 96
- `collections` (*tautulli.models.search_results.EpisodeItem* attribute), 118

collections (tautulli.models.search_results.MovieItem attribute), 123
 collections (tautulli.models.search_results.SeasonItem attribute), 127
 collections (tautulli.models.search_results.ShowItem attribute), 130
 collectionSort (tautulli.models.collections_table.Datum attribute), 73
 CollectionsTable (class in tautulli.models.collections_table), 72
 composite (tautulli.models.playlists_table.Datum attribute), 113
 Config (class in tautulli.models.newsletter_config), 103
 config (tautulli.models.newsletter_config.NewsletterConfig attribute), 106
 config_options (tautulli.models.newsletter_config.NewsletterConfig attribute), 106
 config_version (tautulli.models.settings.Advanced attribute), 138
 config_version (tautulli.models.settings.General attribute), 145
 ConfigOption (class in tautulli.models.newsletter_config), 103
 connected (tautulli.models.server_status.ServerStatus attribute), 137
 container (tautulli.models.activity.Session attribute), 64
 container (tautulli.models.library_media_info.Datum attribute), 90
 container (tautulli.models.metadata.MediaInfoItem attribute), 95
 container (tautulli.models.search_results.MediaInfoItem attribute), 121
 container (tautulli.models.stream_data.StreamData attribute), 178
 container_decision (tautulli.models.activity.Session attribute), 64
 content_rating (tautulli.models.activity.Session attribute), 64
 content_rating (tautulli.models.home_stats.Row attribute), 84
 content_rating (tautulli.models.libraries_table.Datum attribute), 88
 content_rating (tautulli.models.metadata.Metadata attribute), 97
 content_rating (tautulli.models.search_results.EpisodeItem attribute), 118
 content_rating (tautulli.models.search_results.MovieItem attribute), 123
 content_rating (tautulli.models.search_results.SeasonItem attribute), 127
 content_rating (tautulli.models.search_results.ShowItem attribute), 130
 content_type (tautulli.models.synced_items.SyncedItems attribute), 181
 contentRating (tautulli.models.collections_table.Datum attribute), 73
 continent (tautulli.models.geo_ip_lookup.GeoIPLookup attribute), 80
 count (tautulli.models.home_stats.Row attribute), 85
 count (tautulli.models.libraries.LibrariesEntry attribute), 87
 count (tautulli.models.libraries_table.Datum attribute), 88
 count (tautulli.models.library.Library attribute), 86
 country (tautulli.models.geo_ip_lookup.GeoIPLookup attribute), 80
 country (tautulli.models.whois_lookup.Net attribute), 193
 created (tautulli.models.whois_lookup.Net attribute), 193
 cron (tautulli.models.newsletter.Newsletter attribute), 110
 cron (tautulli.models.newsletter_config.NewsletterConfig attribute), 106
 current_session (tautulli.models.stream_data.StreamData attribute), 178
 custom_cron (tautulli.models.newsletter_config.Config attribute), 103

D

data (tautulli.models.collections_table.CollectionsTable attribute), 72
 data (tautulli.models.history.History attribute), 83
 data (tautulli.models.libraries_table.LibrariesTable attribute), 90
 data (tautulli.models.library_media_info.LibraryMediaInfo attribute), 92
 data (tautulli.models.newsletter_log.NewsletterLog attribute), 109
 data (tautulli.models.notification_log.NotificationLog attribute), 111
 data (tautulli.models.playlists_table.PlaylistsTable attribute), 114
 data (tautulli.models.plex_log.PlexLog attribute), 115
 data (tautulli.models.user_ips.UserIPs attribute), 190
 data (tautulli.models.user_logins.UserLogins attribute), 191
 data (tautulli.models.users_table.UsersTable attribute), 188
 date (tautulli.models.history.Datum attribute), 81

`date_format` (*tautulli.models.date_formats.DateFormats* attribute), 74
`date_format` (*tautulli.models.settings.General* attribute), 145
`date_formats` (*tautulli.api.json_api.RawAPI* property), 30
`date_formats` (*tautulli.api.object_api.ObjectAPI* property), 60
`DateFormats` (class in *tautulli.models.date_formats*), 74
`Datum` (class in *tautulli.models.collections_table*), 72
`Datum` (class in *tautulli.models.history*), 81
`Datum` (class in *tautulli.models.libraries_table*), 88
`Datum` (class in *tautulli.models.library_media_info*), 90
`Datum` (class in *tautulli.models.newsletter_log*), 108
`Datum` (class in *tautulli.models.notification_log*), 110
`Datum` (class in *tautulli.models.playlists_table*), 113
`Datum` (class in *tautulli.models.user_ips*), 189
`Datum` (class in *tautulli.models.user_logins*), 190
`Datum` (class in *tautulli.models.users_table*), 186
`delete_all_library_history` (*tautulli.models.docs.Docs* attribute), 74
`delete_all_library_history()` (*tautulli.api.json_api.RawAPI* method), 4
`delete_all_library_history()` (*tautulli.api.object_api.ObjectAPI* method), 34
`delete_all_user_history` (*tautulli.models.docs.Docs* attribute), 74
`delete_all_user_history()` (*tautulli.api.json_api.RawAPI* method), 4
`delete_all_user_history()` (*tautulli.api.object_api.ObjectAPI* method), 34
`delete_cache` (*tautulli.models.docs.Docs* attribute), 74
`delete_cache()` (*tautulli.api.json_api.RawAPI* method), 4
`delete_cache()` (*tautulli.api.object_api.ObjectAPI* method), 34
`delete_export` (*tautulli.models.docs.Docs* attribute), 74
`delete_export()` (*tautulli.api.json_api.RawAPI* method), 4
`delete_export()` (*tautulli.api.object_api.ObjectAPI* method), 34
`delete_history` (*tautulli.models.docs.Docs* attribute), 74
`delete_history()` (*tautulli.api.json_api.RawAPI* method), 5
`delete_history()` (*tautulli.api.object_api.ObjectAPI* method), 35
`delete_hosted_images` (*tautulli.models.docs.Docs* attribute), 74
`delete_hosted_images()` (*tautulli.api.json_api.RawAPI* method), 5
`delete_hosted_images()` (*tautulli.api.object_api.ObjectAPI* method), 35
`delete_image_cache` (*tautulli.models.docs.Docs* attribute), 74
`delete_image_cache()` (*tautulli.api.json_api.RawAPI* method), 5
`delete_image_cache()` (*tautulli.api.object_api.ObjectAPI* method), 35
`delete_library` (*tautulli.models.docs.Docs* attribute), 74
`delete_library()` (*tautulli.api.json_api.RawAPI* method), 5
`delete_library()` (*tautulli.api.object_api.ObjectAPI* method), 35
`delete_login_log` (*tautulli.models.docs.Docs* attribute), 75
`delete_login_log()` (*tautulli.api.json_api.RawAPI* method), 5
`delete_login_log()` (*tautulli.api.object_api.ObjectAPI* method), 35
`delete_lookup_info` (*tautulli.models.docs.Docs* attribute), 75
`delete_lookup_info()` (*tautulli.api.json_api.RawAPI* method), 6
`delete_lookup_info()` (*tautulli.api.object_api.ObjectAPI* method), 36
`delete_media_info_cache` (*tautulli.models.docs.Docs* attribute), 75
`delete_media_info_cache()` (*tautulli.api.json_api.RawAPI* method), 6
`delete_media_info_cache()` (*tautulli.api.object_api.ObjectAPI* method), 36
`delete_mobile_device` (*tautulli.models.docs.Docs* attribute), 75
`delete_mobile_device()` (*tautulli.api.json_api.RawAPI* method), 6
`delete_mobile_device()` (*tautulli.api.object_api.ObjectAPI* method), 36
`delete_newsletter` (*tautulli.models.docs.Docs* attribute), 75
`delete_newsletter()` (*tautulli.api.json_api.RawAPI* method), 6
`delete_newsletter()` (*tautulli.api.object_api.ObjectAPI* method), 36
`delete_newsletter_log` (*tautulli.models.docs.Docs* attribute), 75
`delete_newsletter_log()` (*tautulli.api.json_api.RawAPI* method), 5
`delete_newsletter_log()` (*tautulli.api.object_api.ObjectAPI* method), 36

- tulli.api.json_api.RawAPI* method), 7
- `delete_newsletter_log()` (*tautulli.api.object_api.ObjectAPI* method), 37
- `delete_notification_log` (*tautulli.models.docs.Docs* attribute), 75
- `delete_notification_log()` (*tautulli.api.json_api.RawAPI* method), 7
- `delete_notification_log()` (*tautulli.api.object_api.ObjectAPI* method), 37
- `delete_notifier` (*tautulli.models.docs.Docs* attribute), 75
- `delete_notifier()` (*tautulli.api.json_api.RawAPI* method), 7
- `delete_notifier()` (*tautulli.api.object_api.ObjectAPI* method), 37
- `delete_recently_added` (*tautulli.models.docs.Docs* attribute), 75
- `delete_recently_added()` (*tautulli.api.json_api.RawAPI* method), 7
- `delete_recently_added()` (*tautulli.api.object_api.ObjectAPI* method), 37
- `delete_synced_item` (*tautulli.models.docs.Docs* attribute), 75
- `delete_synced_item()` (*tautulli.api.json_api.RawAPI* method), 7
- `delete_synced_item()` (*tautulli.api.object_api.ObjectAPI* method), 37
- `delete_temp_sessions` (*tautulli.models.docs.Docs* attribute), 75
- `delete_temp_sessions()` (*tautulli.api.json_api.RawAPI* method), 7
- `delete_temp_sessions()` (*tautulli.api.object_api.ObjectAPI* method), 37
- `delete_user` (*tautulli.models.docs.Docs* attribute), 75
- `delete_user()` (*tautulli.api.json_api.RawAPI* method), 8
- `delete_user()` (*tautulli.api.object_api.ObjectAPI* method), 38
- `deleted_section` (*tautulli.models.library.Library* attribute), 86
- `deleted_user` (*tautulli.models.activity.Session* attribute), 64
- `deleted_user` (*tautulli.models.user.User* attribute), 184
- `description` (*tautulli.models.newsletter_config.ConfigOptional* attribute), 103
- `description` (*tautulli.models.newsletter_config.EmailConfigOptional* attribute), 105
- `description` (*tautulli.models.whois_lookup.Net* attribute), 193
- `device` (*tautulli.models.activity.Session* attribute), 64
- `device_name` (*tautulli.models.synced_items.SyncedItems* attribute), 181
- `directors` (*tautulli.models.activity.Session* attribute), 64
- `directors` (*tautulli.models.metadata.Metadata* attribute), 97
- `directors` (*tautulli.models.search_results.EpisodeItem* attribute), 118
- `directors` (*tautulli.models.search_results.MovieItem* attribute), 123
- `directors` (*tautulli.models.search_results.SeasonItem* attribute), 127
- `directors` (*tautulli.models.search_results.ShowItem* attribute), 130
- `distro` (*tautulli.models.pms_update.PMSUpdate* attribute), 115
- `distro_build` (*tautulli.models.pms_update.PMSUpdate* attribute), 115
- `do_not_override_git_branch` (*tautulli.models.settings.General* attribute), 145
- `do_notify` (*tautulli.models.activity.Session* attribute), 64
- `do_notify` (*tautulli.models.libraries_table.Datum* attribute), 88
- `do_notify` (*tautulli.models.library.Library* attribute), 86
- `do_notify` (*tautulli.models.user.User* attribute), 184
- `do_notify` (*tautulli.models.users.User* attribute), 185
- `do_notify` (*tautulli.models.users_table.Datum* attribute), 186
- `do_notify_created` (*tautulli.models.libraries_table.Datum* attribute), 88
- `do_notify_created` (*tautulli.models.library.Library* attribute), 86
- `Docs` (class in *tautulli.models.docs*), 74
- `docs` (*tautulli.api.json_api.RawAPI* property), 30
- `docs` (*tautulli.api.object_api.ObjectAPI* property), 60
- `docs` (*tautulli.models.docs.Docs* attribute), 75
- `docs_md` (*tautulli.api.json_api.RawAPI* property), 30
- `docs_md` (*tautulli.api.object_api.ObjectAPI* property), 60
- `docs_md` (*tautulli.models.docs.Docs* attribute), 75
- `download_config` (*tautulli.models.docs.Docs* attribute), 75
- `download_config()` (*tautulli.api.json_api.RawAPI* method), 8
- `download_config()` (*tautulli.api.object_api.ObjectAPI* method), 38
- `download_database` (*tautulli.models.docs.Docs* attribute), 75
- `download_database()` (*tautulli.api.json_api.RawAPI* method), 8
- `download_database()` (*tautulli.api.object_api.ObjectAPI* method), 38
- `download_export` (*tautulli.models.docs.Docs* attribute),

- 75
`download_export()` (*tautulli.api.json_api.RawAPI method*), 8
`download_export()` (*tautulli.api.object_api.ObjectAPI method*), 38
`download_log` (*tautulli.models.docs.Docs attribute*), 75
`download_log()` (*tautulli.api.json_api.RawAPI method*), 8
`download_log()` (*tautulli.api.object_api.ObjectAPI method*), 38
`download_plex_log` (*tautulli.models.docs.Docs attribute*), 75
`download_plex_log()` (*tautulli.api.json_api.RawAPI method*), 8
`download_plex_log()` (*tautulli.api.object_api.ObjectAPI method*), 38
`download_url` (*tautulli.models.pms_update.PMSUpdate attribute*), 115
`draw` (*tautulli.models.collections_table.CollectionsTable attribute*), 72
`draw` (*tautulli.models.history.History attribute*), 83
`draw` (*tautulli.models.libraries_table.LibrariesTable attribute*), 90
`draw` (*tautulli.models.library_media_info.LibraryMediaInfo attribute*), 92
`draw` (*tautulli.models.newsletter_log.NewsletterLog attribute*), 109
`draw` (*tautulli.models.notification_log.NotificationLog attribute*), 111
`draw` (*tautulli.models.playlists_table.PlaylistsTable attribute*), 114
`draw` (*tautulli.models.user_ips.UserIPs attribute*), 190
`draw` (*tautulli.models.user_logins.UserLogins attribute*), 191
`draw` (*tautulli.models.users_table.UsersTable attribute*), 188
`duration` (*tautulli.models.activity.Session attribute*), 64
`duration` (*tautulli.models.history.Datum attribute*), 81
`duration` (*tautulli.models.libraries_table.Datum attribute*), 88
`duration` (*tautulli.models.metadata.Metadata attribute*), 97
`duration` (*tautulli.models.playlists_table.Datum attribute*), 113
`duration` (*tautulli.models.search_results.EpisodeItem attribute*), 118
`duration` (*tautulli.models.search_results.MovieItem attribute*), 123
`duration` (*tautulli.models.search_results.SeasonItem attribute*), 127
`duration` (*tautulli.models.search_results.ShowItem attribute*), 130
`duration` (*tautulli.models.users_table.Datum attribute*), 186
`duration_milliseconds` (*tautulli.models.activity.Session property*), 64
- ## E
- `edit_library` (*tautulli.models.docs.Docs attribute*), 75
`edit_library()` (*tautulli.api.json_api.RawAPI method*), 9
`edit_library()` (*tautulli.api.object_api.ObjectAPI method*), 38
`edit_user` (*tautulli.models.docs.Docs attribute*), 75
`edit_user()` (*tautulli.api.json_api.RawAPI method*), 9
`edit_user()` (*tautulli.api.object_api.ObjectAPI method*), 39
`edition_title` (*tautulli.models.metadata.Metadata attribute*), 97
`Email` (*class in tautulli.models.settings*), 141
`email` (*tautulli.models.activity.Session attribute*), 64
`Email` (*tautulli.models.settings.Settings attribute*), 170
`email` (*tautulli.models.user.User attribute*), 184
`email` (*tautulli.models.users.User attribute*), 185
`email` (*tautulli.models.users_table.Datum attribute*), 186
`email_bcc` (*tautulli.models.settings.Email attribute*), 141
`email_cc` (*tautulli.models.settings.Email attribute*), 141
`email_config` (*tautulli.models.newsletter_config.NewsletterConfig attribute*), 106
`email_config_options` (*tautulli.models.newsletter_config.NewsletterConfig attribute*), 106
`email_enabled` (*tautulli.models.settings.Email attribute*), 141
`email_from` (*tautulli.models.settings.Email attribute*), 141
`email_from_name` (*tautulli.models.settings.Email attribute*), 142
`email_html_support` (*tautulli.models.settings.Email attribute*), 142
`email_on_buffer` (*tautulli.models.settings.Email attribute*), 142
`email_on_concurrent` (*tautulli.models.settings.Email attribute*), 142
`email_on_created` (*tautulli.models.settings.Email attribute*), 142
`email_on_extdown` (*tautulli.models.settings.Email attribute*), 142
`email_on_extup` (*tautulli.models.settings.Email attribute*), 142
`email_on_intdown` (*tautulli.models.settings.Email attribute*), 142
`email_on_intup` (*tautulli.models.settings.Email attribute*), 142
`email_on_newdevice` (*tautulli.models.settings.Email attribute*), 142

email_on_pause (*tautulli.models.settings.Email attribute*), 142
 email_on_play (*tautulli.models.settings.Email attribute*), 142
 email_on_pmsupdate (*tautulli.models.settings.Email attribute*), 142
 email_on_resume (*tautulli.models.settings.Email attribute*), 142
 email_on_stop (*tautulli.models.settings.Email attribute*), 142
 email_on_watched (*tautulli.models.settings.Email attribute*), 142
 email_smtp_password (*tautulli.models.settings.Email attribute*), 142
 email_smtp_port (*tautulli.models.settings.Email attribute*), 142
 email_smtp_server (*tautulli.models.settings.Email attribute*), 142
 email_smtp_user (*tautulli.models.settings.Email attribute*), 142
 email_tls (*tautulli.models.settings.Email attribute*), 142
 email_to (*tautulli.models.settings.Email attribute*), 142
 EmailConfig (class in *tautulli.models.newsletter_config*), 104
 EmailConfigOption (class in *tautulli.models.newsletter_config*), 104
 emails (*tautulli.models.whois_lookup.Net attribute*), 193
 enable_https (*tautulli.models.settings.General attribute*), 145
 end_date (*tautulli.models.newsletter_log.Datum attribute*), 108
 end_time_offset (*tautulli.models.metadata.Marker attribute*), 94
 episode (*tautulli.models.search_results.ResultsList attribute*), 126
 EpisodeItem (class in *tautulli.models.search_results*), 118
 eta (*tautulli.models.activity.Session property*), 64
 export_dir (*tautulli.models.settings.General attribute*), 145
 export_metadata (*tautulli.models.docs.Docs attribute*), 75
 export_metadata() (*tautulli.api.json_api.RawAPI method*), 9
 export_metadata() (*tautulli.api.object_api.ObjectAPI method*), 39
 export_threads (*tautulli.models.settings.Advanced attribute*), 138
 ExportFields (class in *tautulli.models.export_fields*), 79
 extra_info (*tautulli.models.pms_update.PMSUpdate attribute*), 115

F

Facebook (class in *tautulli.models.settings*), 143
 Facebook (*tautulli.models.settings.Settings attribute*), 170
 facebook_app_id (*tautulli.models.settings.Facebook attribute*), 143
 facebook_app_secret (*tautulli.models.settings.Facebook attribute*), 143
 facebook_enabled (*tautulli.models.settings.Facebook attribute*), 143
 facebook_group (*tautulli.models.settings.Facebook attribute*), 143
 facebook_incl_pmslink (*tautulli.models.settings.Facebook attribute*), 143
 facebook_incl_poster (*tautulli.models.settings.Facebook attribute*), 143
 facebook_incl_subject (*tautulli.models.settings.Facebook attribute*), 143
 facebook_on_buffer (*tautulli.models.settings.Facebook attribute*), 143
 facebook_on_concurrent (*tautulli.models.settings.Facebook attribute*), 143
 facebook_on_created (*tautulli.models.settings.Facebook attribute*), 143
 facebook_on_extdown (*tautulli.models.settings.Facebook attribute*), 143
 facebook_on_extup (*tautulli.models.settings.Facebook attribute*), 143
 facebook_on_intdown (*tautulli.models.settings.Facebook attribute*), 143
 facebook_on_intup (*tautulli.models.settings.Facebook attribute*), 144
 facebook_on_newdevice (*tautulli.models.settings.Facebook attribute*), 144
 facebook_on_pause (*tautulli.models.settings.Facebook attribute*), 144
 facebook_on_play (*tautulli.models.settings.Facebook attribute*), 144
 facebook_on_pmsupdate (*tautulli.models.settings.Facebook attribute*), 144
 facebook_on_resume (*tautulli.models.settings.Facebook attribute*), 144

facebook_on_stop (*tautulli.models.settings.Facebook attribute*), 144
 facebook_on_watched (*tautulli.models.settings.Facebook attribute*), 144
 facebook_redirect_uri (*tautulli.models.settings.Facebook attribute*), 144
 facebook_token (*tautulli.models.settings.Facebook attribute*), 144
 failure (*tautulli.models.synced_items.SyncedItems attribute*), 181
 field (*tautulli.models.export_fields.MediaInfoField attribute*), 80
 field (*tautulli.models.export_fields.MetadataField attribute*), 80
 Field0 (*class in tautulli.models.new_rating_keys*), 102
 Field0 (*class in tautulli.models.old_rating_keys*), 113
 field_0 (*tautulli.models.new_rating_keys.NewRatingKeys attribute*), 102
 field_0 (*tautulli.models.old_rating_keys.OldRatingKeys attribute*), 113
 file (*tautulli.models.activity.Session attribute*), 64
 file (*tautulli.models.metadata.Part attribute*), 99
 file (*tautulli.models.search_results.Part attribute*), 125
 file_size (*tautulli.models.activity.Session attribute*), 64
 file_size (*tautulli.models.library_media_info.Datum attribute*), 90
 file_size (*tautulli.models.metadata.Part attribute*), 99
 file_size (*tautulli.models.search_results.Part attribute*), 125
 filename (*tautulli.models.newsletter_config.Config attribute*), 103
 filter_all (*tautulli.models.users.User attribute*), 185
 filter_duration (*tautulli.models.history.History attribute*), 84
 filter_movies (*tautulli.models.users.User attribute*), 185
 filter_music (*tautulli.models.users.User attribute*), 185
 filter_photos (*tautulli.models.users.User attribute*), 185
 filter_tv (*tautulli.models.users.User attribute*), 185
 filtered_file_size (*tautulli.models.library_media_info.LibraryMediaInfo attribute*), 92
 final (*tautulli.models.metadata.Marker attribute*), 94
 first (*tautulli.models.metadata.Marker attribute*), 94
 first_run_complete (*tautulli.models.settings.General attribute*), 145
 first_seen (*tautulli.models.user_ips.Datum attribute*), 189
 formatted (*tautulli.models.newsletter_config.Config attribute*), 103
 freeze_db (*tautulli.models.settings.General attribute*), 145
 friendly_name (*tautulli.models.activity.Session attribute*), 64
 friendly_name (*tautulli.models.history.Datum attribute*), 81
 friendly_name (*tautulli.models.home_stats.Row attribute*), 85
 friendly_name (*tautulli.models.library_user_stats.LibraryUserStats attribute*), 93
 friendly_name (*tautulli.models.newsletter.Newsletter attribute*), 110
 friendly_name (*tautulli.models.newsletter_config.NewsletterConfig attribute*), 106
 friendly_name (*tautulli.models.notifiers.Notifier attribute*), 112
 friendly_name (*tautulli.models.user.User attribute*), 184
 friendly_name (*tautulli.models.user_ips.Datum attribute*), 189
 friendly_name (*tautulli.models.user_logins.Datum attribute*), 190
 friendly_name (*tautulli.models.usernames.UserName attribute*), 192
 friendly_name (*tautulli.models.users.User attribute*), 185
 friendly_name (*tautulli.models.users_table.Datum attribute*), 186
 from_ (*tautulli.models.newsletter_config.EmailConfig attribute*), 104
 from_name (*tautulli.models.newsletter_config.EmailConfig attribute*), 104
 full_title (*tautulli.models.activity.Session attribute*), 64
 full_title (*tautulli.models.history.Datum attribute*), 81
 full_title (*tautulli.models.metadata.Metadata attribute*), 97
 full_title (*tautulli.models.search_results.EpisodeItem attribute*), 118
 full_title (*tautulli.models.search_results.MovieItem attribute*), 123
 full_title (*tautulli.models.search_results.SeasonItem attribute*), 127
 full_title (*tautulli.models.search_results.ShowItem attribute*), 130

G

General (*class in tautulli.models.settings*), 144
 General (*tautulli.models.settings.Settings attribute*), 170
 genres (*tautulli.models.activity.Session attribute*), 64
 genres (*tautulli.models.metadata.Metadata attribute*), 97

`genres` (`tautulli.models.search_results.EpisodeItem` attribute), 118

`genres` (`tautulli.models.search_results.MovieItem` attribute), 123

`genres` (`tautulli.models.search_results.SeasonItem` attribute), 127

`genres` (`tautulli.models.search_results.ShowItem` attribute), 130

`geoip_db` (`tautulli.models.settings.General` attribute), 145

`geoip_db_installed` (`tautulli.models.settings.General` attribute), 145

`geoip_db_update_days` (`tautulli.models.settings.General` attribute), 145

`GeoIPLookup` (class in `tautulli.models.geo_ip_lookup`), 80

`get_activity` (`tautulli.models.docs.Docs` attribute), 75

`get_api_key()` (`tautulli.api.json_api.RawAPI` method), 10

`get_api_key()` (`tautulli.api.object_api.ObjectAPI` method), 40

`get_apikey` (`tautulli.models.docs.Docs` attribute), 75

`get_children_metadata()` (`tautulli.api.json_api.RawAPI` method), 10

`get_children_metadata()` (`tautulli.api.object_api.ObjectAPI` method), 40

`get_collections_table` (`tautulli.models.docs.Docs` attribute), 75

`get_collections_table()` (`tautulli.api.json_api.RawAPI` method), 10

`get_collections_table()` (`tautulli.api.object_api.ObjectAPI` method), 40

`get_date_formats` (`tautulli.models.docs.Docs` attribute), 75

`get_export_fields` (`tautulli.models.docs.Docs` attribute), 75

`get_export_fields()` (`tautulli.api.json_api.RawAPI` method), 10

`get_export_fields()` (`tautulli.api.object_api.ObjectAPI` method), 40

`get_exports_table` (`tautulli.models.docs.Docs` attribute), 75

`get_exports_table()` (`tautulli.api.json_api.RawAPI` method), 11

`get_exports_table()` (`tautulli.api.object_api.ObjectAPI` method), 41

`get_file_sizes` (`tautulli.models.settings.General` attribute), 145

`get_file_sizes_hold` (`tautulli.models.settings.General` attribute), 145

`get_geoip_lookup` (`tautulli.models.docs.Docs` attribute), 75

`get_geoip_lookup()` (`tautulli.api.json_api.RawAPI` method), 11

`get_geoip_lookup()` (`tautulli.api.object_api.ObjectAPI` method), 41

`get_history` (`tautulli.models.docs.Docs` attribute), 75

`get_history()` (`tautulli.api.json_api.RawAPI` method), 11

`get_history()` (`tautulli.api.object_api.ObjectAPI` method), 41

`get_home_stats` (`tautulli.models.docs.Docs` attribute), 75

`get_home_stats()` (`tautulli.api.json_api.RawAPI` method), 12

`get_home_stats()` (`tautulli.api.object_api.ObjectAPI` method), 42

`get_item_user_stats()` (`tautulli.api.json_api.RawAPI` method), 13

`get_item_user_stats()` (`tautulli.api.object_api.ObjectAPI` method), 42

`get_item_watch_time_stats()` (`tautulli.api.json_api.RawAPI` method), 13

`get_item_watch_time_stats()` (`tautulli.api.object_api.ObjectAPI` method), 43

`get_libraries` (`tautulli.models.docs.Docs` attribute), 76

`get_libraries_table` (`tautulli.models.docs.Docs` attribute), 76

`get_libraries_table()` (`tautulli.api.json_api.RawAPI` method), 13

`get_libraries_table()` (`tautulli.api.object_api.ObjectAPI` method), 43

`get_library` (`tautulli.models.docs.Docs` attribute), 76

`get_library()` (`tautulli.api.json_api.RawAPI` method), 13

`get_library()` (`tautulli.api.object_api.ObjectAPI` method), 43

`get_library_by_name()` (`tautulli.tools.api_helper.APIShortcuts` method), 195

`get_library_media_info` (`tautulli.models.docs.Docs` attribute), 76

`get_library_media_info()` (`tautulli.api.json_api.RawAPI` method), 14

`get_library_media_info()` (`tautulli.api.object_api.ObjectAPI` method), 43

`get_library_names` (`tautulli.models.docs.Docs` attribute), 76

`get_library_user_stats` (*tautulli.models.docs.Docs attribute*), 76
`get_library_user_stats()` (*tautulli.api.json_api.RawAPI method*), 14
`get_library_user_stats()` (*tautulli.api.object_api.ObjectAPI method*), 44
`get_library_watch_time_stats` (*tautulli.models.docs.Docs attribute*), 76
`get_library_watch_time_stats()` (*tautulli.api.json_api.RawAPI method*), 14
`get_library_watch_time_stats()` (*tautulli.api.object_api.ObjectAPI method*), 44
`get_logs` (*tautulli.models.docs.Docs attribute*), 76
`get_logs()` (*tautulli.api.json_api.RawAPI method*), 15
`get_logs()` (*tautulli.api.object_api.ObjectAPI method*), 44
`get_metadata` (*tautulli.models.docs.Docs attribute*), 76
`get_metadata()` (*tautulli.api.json_api.RawAPI method*), 15
`get_metadata()` (*tautulli.api.object_api.ObjectAPI method*), 45
`get_new_rating_keys` (*tautulli.models.docs.Docs attribute*), 76
`get_new_rating_keys()` (*tautulli.api.json_api.RawAPI method*), 15
`get_new_rating_keys()` (*tautulli.api.object_api.ObjectAPI method*), 45
`get_newsletter_config` (*tautulli.models.docs.Docs attribute*), 76
`get_newsletter_config()` (*tautulli.api.json_api.RawAPI method*), 15
`get_newsletter_config()` (*tautulli.api.object_api.ObjectAPI method*), 45
`get_newsletter_log` (*tautulli.models.docs.Docs attribute*), 76
`get_newsletter_log()` (*tautulli.api.json_api.RawAPI method*), 16
`get_newsletter_log()` (*tautulli.api.object_api.ObjectAPI method*), 45
`get_newsletters` (*tautulli.models.docs.Docs attribute*), 76
`get_notification_log` (*tautulli.models.docs.Docs attribute*), 76
`get_notification_log()` (*tautulli.api.json_api.RawAPI method*), 16
`get_notification_log()` (*tautulli.api.object_api.ObjectAPI method*), 46
`get_notifier_config` (*tautulli.models.docs.Docs attribute*), 76
`get_notifier_config()` (*tautulli.api.json_api.RawAPI method*), 16
`get_notifier_config()` (*tautulli.api.object_api.ObjectAPI method*), 46
`get_notifier_parameters` (*tautulli.models.docs.Docs attribute*), 76
`get_notifiers` (*tautulli.models.docs.Docs attribute*), 76
`get_notifiers()` (*tautulli.api.json_api.RawAPI method*), 17
`get_notifiers()` (*tautulli.api.object_api.ObjectAPI method*), 46
`get_old_rating_keys` (*tautulli.models.docs.Docs attribute*), 76
`get_old_rating_keys()` (*tautulli.api.json_api.RawAPI method*), 17
`get_old_rating_keys()` (*tautulli.api.object_api.ObjectAPI method*), 47
`get_playlists_table` (*tautulli.models.docs.Docs attribute*), 76
`get_playlists_table()` (*tautulli.api.json_api.RawAPI method*), 17
`get_playlists_table()` (*tautulli.api.object_api.ObjectAPI method*), 47
`get_plays_by_date` (*tautulli.models.docs.Docs attribute*), 76
`get_plays_by_date()` (*tautulli.api.json_api.RawAPI method*), 17
`get_plays_by_date()` (*tautulli.api.object_api.ObjectAPI method*), 47
`get_plays_by_day_of_week()` (*tautulli.api.json_api.RawAPI method*), 17
`get_plays_by_day_of_week()` (*tautulli.api.object_api.ObjectAPI method*), 47
`get_plays_by_dayofweek` (*tautulli.models.docs.Docs attribute*), 76
`get_plays_by_hour_of_day()` (*tautulli.api.json_api.RawAPI method*), 18
`get_plays_by_hour_of_day()` (*tautulli.api.object_api.ObjectAPI method*), 48
`get_plays_by_hourofday` (*tautulli.models.docs.Docs attribute*), 76
`get_plays_by_source_resolution` (*tautulli.models.docs.Docs attribute*), 76
`get_plays_by_source_resolution()` (*tautulli.api.json_api.RawAPI method*), 18
`get_plays_by_source_resolution()` (*tautulli.api.object_api.ObjectAPI method*), 48

<i>tulli.api.object_api.ObjectAPI</i>	<i>method</i>),	77
48		
<code>get_plays_by_stream_resolution</code>	(<i>tautulli.models.docs.Docs</i> attribute), 76	
<code>get_plays_by_stream_resolution()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 18	
<code>get_plays_by_stream_resolution()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method),	
48		
<code>get_plays_by_stream_type</code>	(<i>tautulli.models.docs.Docs</i> attribute), 76	
<code>get_plays_by_stream_type()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 19	
<code>get_plays_by_stream_type()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method),	
48		
<code>get_plays_by_top_10_platforms</code>	(<i>tautulli.models.docs.Docs</i> attribute), 76	
<code>get_plays_by_top_10_platforms()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 19	
<code>get_plays_by_top_10_platforms()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method),	
49		
<code>get_plays_by_top_10_users</code>	(<i>tautulli.models.docs.Docs</i> attribute), 76	
<code>get_plays_by_top_10_users()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 19	
<code>get_plays_by_top_10_users()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method),	
49		
<code>get_plays_per_month</code> (<i>tautulli.models.docs.Docs</i> attribute), 76		
<code>get_plays_per_month()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 19	
<code>get_plays_per_month()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method),	
49		
<code>get_plex_log</code> (<i>tautulli.models.docs.Docs</i> attribute), 76		
<code>get_plex_log()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 20	
<code>get_plex_log()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method), 50	
<code>get_pms_update</code> (<i>tautulli.models.docs.Docs</i> attribute), 76		
<code>get_recently_added</code>	(<i>tautulli.models.docs.Docs</i> attribute), 76	
<code>get_recently_added()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 20	
<code>get_recently_added()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method),	
50		
<code>get_server_friendly_name</code>	(<i>tautulli.models.docs.Docs</i> attribute), 77	
<code>get_server_id</code> (<i>tautulli.models.docs.Docs</i> attribute),		
		77
<code>get_server_id()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 20	
<code>get_server_id()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method), 50	
<code>get_server_identity</code> (<i>tautulli.models.docs.Docs</i> attribute), 77		
<code>get_server_info</code> (<i>tautulli.models.docs.Docs</i> attribute),		
77		
<code>get_server_list</code> (<i>tautulli.models.docs.Docs</i> attribute),		
77		
<code>get_server_pref</code> (<i>tautulli.models.docs.Docs</i> attribute),		
77		
<code>get_server_pref()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 21	
<code>get_server_pref()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method), 50	
<code>get_servers_info</code> (<i>tautulli.models.docs.Docs</i> attribute), 77		
<code>get_settings</code> (<i>tautulli.models.docs.Docs</i> attribute), 77		
<code>get_settings()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 21	
<code>get_settings()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method), 51	
<code>get_stream_data</code> (<i>tautulli.models.docs.Docs</i> attribute),		
77		
<code>get_stream_data()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 21	
<code>get_stream_data()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method), 51	
<code>get_stream_type_by_top_10_platforms</code>	(<i>tautulli.models.docs.Docs</i> attribute), 77	
<code>get_stream_type_by_top_10_platforms()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 21	
<code>get_stream_type_by_top_10_platforms()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method), 51	
<code>get_stream_type_by_top_10_users</code>	(<i>tautulli.models.docs.Docs</i> attribute), 77	
<code>get_stream_type_by_top_10_users()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 21	
<code>get_stream_type_by_top_10_users()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method),	
51		
<code>get_synced_items</code> (<i>tautulli.models.docs.Docs</i> attribute), 77		
<code>get_synced_items()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 22	
<code>get_synced_items()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method),	
52		
<code>get_user</code> (<i>tautulli.models.docs.Docs</i> attribute), 77		
<code>get_user()</code>	(<i>tautulli.api.json_api.RawAPI</i> method), 22	
<code>get_user()</code>	(<i>tautulli.api.object_api.ObjectAPI</i> method),	
52		

`get_user_ips` (*tautulli.models.docs.Docs* attribute), 77
`get_user_ips()` (*tautulli.api.json_api.RawAPI* method), 22
`get_user_ips()` (*tautulli.api.object_api.ObjectAPI* method), 52
`get_user_logins` (*tautulli.models.docs.Docs* attribute), 77
`get_user_logins()` (*tautulli.api.json_api.RawAPI* method), 23
`get_user_logins()` (*tautulli.api.object_api.ObjectAPI* method), 52
`get_user_names` (*tautulli.models.docs.Docs* attribute), 77
`get_user_player_stats` (*tautulli.models.docs.Docs* attribute), 77
`get_user_player_stats()` (*tautulli.api.json_api.RawAPI* method), 23
`get_user_player_stats()` (*tautulli.api.object_api.ObjectAPI* method), 53
`get_user_watch_time_stats` (*tautulli.models.docs.Docs* attribute), 77
`get_user_watch_time_stats()` (*tautulli.api.json_api.RawAPI* method), 23
`get_user_watch_time_stats()` (*tautulli.api.object_api.ObjectAPI* method), 53
`get_users` (*tautulli.models.docs.Docs* attribute), 77
`get_users_table` (*tautulli.models.docs.Docs* attribute), 77
`get_users_table()` (*tautulli.api.json_api.RawAPI* method), 23
`get_users_table()` (*tautulli.api.object_api.ObjectAPI* method), 53
`get_whois_lookup` (*tautulli.models.docs.Docs* attribute), 77
`get_whois_lookup()` (*tautulli.api.json_api.RawAPI* method), 24
`get_whois_lookup()` (*tautulli.api.object_api.ObjectAPI* method), 54
`GetFileSizesHold` (class in *tautulli.models.settings*), 149
`git_branch` (*tautulli.models.settings.General* attribute), 145
`git_path` (*tautulli.models.settings.General* attribute), 145
`git_remote` (*tautulli.models.settings.General* attribute), 145
`git_repo` (*tautulli.models.settings.General* attribute), 145
`git_token` (*tautulli.models.settings.General* attribute), 145
`git_user` (*tautulli.models.settings.General* attribute), 145
`grandparent_guid` (*tautulli.models.activity.Session* attribute), 64
`grandparent_guid` (*tautulli.models.metadata.Metadata* attribute), 97
`grandparent_guid` (*tautulli.models.search_results.EpisodeItem* attribute), 118
`grandparent_guid` (*tautulli.models.search_results.MovieItem* attribute), 123
`grandparent_guid` (*tautulli.models.search_results.SeasonItem* attribute), 127
`grandparent_guid` (*tautulli.models.search_results.ShowItem* attribute), 130
`grandparent_guids` (*tautulli.models.metadata.Metadata* attribute), 97
`grandparent_rating_key` (*tautulli.models.activity.Session* attribute), 64
`grandparent_rating_key` (*tautulli.models.history.Datum* attribute), 81
`grandparent_rating_key` (*tautulli.models.library_media_info.Datum* attribute), 90
`grandparent_rating_key` (*tautulli.models.metadata.Metadata* attribute), 97
`grandparent_rating_key` (*tautulli.models.search_results.EpisodeItem* attribute), 118
`grandparent_rating_key` (*tautulli.models.search_results.MovieItem* attribute), 123
`grandparent_rating_key` (*tautulli.models.search_results.SeasonItem* attribute), 127
`grandparent_rating_key` (*tautulli.models.search_results.ShowItem* attribute), 130
`grandparent_thumb` (*tautulli.models.activity.Session* attribute), 65
`grandparent_thumb` (*tautulli.models.home_stats.Row* attribute), 85
`grandparent_thumb` (*tautulli.models.metadata.Metadata* attribute), 97
`grandparent_thumb` (*tautulli.models.search_results.EpisodeItem* attribute), 118
`grandparent_thumb` (*tautulli.models.search_results.MovieItem* attribute), 118

tribute), 123		growl_enabled (tautulli.models.settings.Growl attribute), 149
grandparent_thumb (tautulli.models.search_results.SeasonItem attribute), 127		growl_host (tautulli.models.settings.Growl attribute), 149
grandparent_thumb (tautulli.models.search_results.ShowItem attribute), 130		growl_on_buffer (tautulli.models.settings.Growl attribute), 150
grandparent_title (tautulli.models.activity.Session attribute), 65		growl_on_concurrent (tautulli.models.settings.Growl attribute), 150
grandparent_title (tautulli.models.history.Datum attribute), 81		growl_on_created (tautulli.models.settings.Growl attribute), 150
grandparent_title (tautulli.models.metadata.Metadata attribute), 97		growl_on_extdown (tautulli.models.settings.Growl attribute), 150
grandparent_title (tautulli.models.search_results.EpisodeItem attribute), 118		growl_on_extup (tautulli.models.settings.Growl attribute), 150
grandparent_title (tautulli.models.search_results.MovieItem attribute), 123		growl_on_intdown (tautulli.models.settings.Growl attribute), 150
grandparent_title (tautulli.models.search_results.SeasonItem attribute), 127		growl_on_intup (tautulli.models.settings.Growl attribute), 150
grandparent_title (tautulli.models.search_results.ShowItem attribute), 130		growl_on_newdevice (tautulli.models.settings.Growl attribute), 150
grandparent_title (tautulli.models.stream_data.StreamData attribute), 178		growl_on_pause (tautulli.models.settings.Growl attribute), 150
grandparent_year (tautulli.models.metadata.Metadata attribute), 97		growl_on_play (tautulli.models.settings.Growl attribute), 150
graph_days (tautulli.models.settings.General attribute), 146		growl_on_pmsupdate (tautulli.models.settings.Growl attribute), 150
graph_months (tautulli.models.settings.General attribute), 146		growl_on_resume (tautulli.models.settings.Growl attribute), 150
graph_tab (tautulli.models.settings.General attribute), 146		growl_on_stop (tautulli.models.settings.Growl attribute), 150
graph_type (tautulli.models.settings.General attribute), 146		growl_on_watched (tautulli.models.settings.Growl attribute), 150
group_count (tautulli.models.history.Datum attribute), 81		growl_password (tautulli.models.settings.Growl attribute), 150
group_history_tables (tautulli.models.settings.General attribute), 146		guid (tautulli.models.activity.Session attribute), 65
group_ids (tautulli.models.history.Datum attribute), 81		guid (tautulli.models.collections_table.Datum attribute), 73
grouping_charts (tautulli.models.settings.PlexWatch attribute), 163		guid (tautulli.models.history.Datum attribute), 81
grouping_global_history (tautulli.models.settings.PlexWatch attribute), 163		guid (tautulli.models.home_stats.Row attribute), 85
grouping_user_history (tautulli.models.settings.PlexWatch attribute), 163		guid (tautulli.models.libraries_table.Datum attribute), 88
Growl (class in tautulli.models.settings), 149		guid (tautulli.models.metadata.Metadata attribute), 97
Growl (tautulli.models.settings.Settings attribute), 171		guid (tautulli.models.playlists_table.Datum attribute), 113
		guid (tautulli.models.search_results.EpisodeItem attribute), 118
		guid (tautulli.models.search_results.MovieItem attribute), 123
		guid (tautulli.models.search_results.SeasonItem attribute), 127
		guid (tautulli.models.search_results.ShowItem attribute), 130
		guid (tautulli.models.user_ips.Datum attribute), 189
		guid (tautulli.models.users_table.Datum attribute), 186

- guids (*tautulli.models.activity.Session* attribute), 65
- guids (*tautulli.models.metadata.Metadata* attribute), 97
- guids (*tautulli.models.search_results.EpisodeItem* attribute), 118
- guids (*tautulli.models.search_results.MovieItem* attribute), 123
- guids (*tautulli.models.search_results.SeasonItem* attribute), 127
- guids (*tautulli.models.search_results.ShowItem* attribute), 130
- ## H
- handle (*tautulli.models.whois_lookup.Net* attribute), 193
- has_plex_pass (*tautulli.tools.api_helper.APIShortcuts* property), 195
- height (*tautulli.models.activity.Session* attribute), 65
- height (*tautulli.models.metadata.MediaInfoItem* attribute), 95
- height (*tautulli.models.search_results.MediaInfoItem* attribute), 121
- Hipchat (class in *tautulli.models.settings*), 150
- Hipchat (*tautulli.models.settings.Settings* attribute), 171
- hipchat_color (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_emoticon (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_enabled (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_incl_pmslink (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_incl_poster (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_incl_subject (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_buffer (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_concurrent (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_created (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_extdown (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_extup (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_intdown (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_intup (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_newdevice (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_pause (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_play (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_pmsupdate (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_resume (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_stop (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_on_watched (*tautulli.models.settings.Hipchat* attribute), 151
- hipchat_url (*tautulli.models.settings.Hipchat* attribute), 151
- History (class in *tautulli.models.history*), 83
- history_row_id (*tautulli.models.libraries_table.Datum* attribute), 88
- history_row_id (*tautulli.models.users_table.Datum* attribute), 186
- history_table_activity (*tautulli.models.settings.General* attribute), 146
- home_library_cards (*tautulli.models.settings.General* attribute), 146
- home_refresh_interval (*tautulli.models.settings.General* attribute), 146
- home_sections (*tautulli.models.settings.General* attribute), 146
- home_stats_cards (*tautulli.models.settings.General* attribute), 146
- home_stats_count (*tautulli.models.settings.General* attribute), 146
- home_stats_length (*tautulli.models.settings.General* attribute), 146
- home_stats_recently_added_count (*tautulli.models.settings.General* attribute), 146
- home_stats_type (*tautulli.models.settings.General* attribute), 146
- HomeStat (class in *tautulli.models.home_stats*), 84
- host (*tautulli.models.servers_info.ServersInfoEntry* attribute), 137
- host (*tautulli.models.user_logins.Datum* attribute), 190
- host (*tautulli.models.whois_lookup.WHOISLookup* attribute), 194
- html_support (*tautulli.models.newsletter_config.EmailConfig* attribute), 104
- http_base_url (*tautulli.models.settings.General* attribute), 146
- http_basic_auth (*tautulli.models.settings.General* attribute), 146
- http_environment (*tautulli.models.settings.General* attribute), 146
- http_hash_password (*tautulli.models.settings.General* attribute), 146
- http_hashed_password (*tautulli.models.settings.General* attribute), 146
- http_host (*tautulli.models.settings.General* attribute), 146
- http_password (*tautulli.models.settings.General* attribute), 146

attribute), 146
 http_plex_admin (tautulli.models.settings.General attribute), 146
 http_port (tautulli.models.settings.General attribute), 146
 http_proxy (tautulli.models.settings.General attribute), 146
 http_rate_limit_attempts (tautulli.models.settings.General attribute), 146
 http_rate_limit_attempts_interval (tautulli.models.settings.General attribute), 146
 http_rate_limit_lockout_time (tautulli.models.settings.General attribute), 146
 http_root (tautulli.models.settings.General attribute), 146
 http_username (tautulli.models.settings.General attribute), 146
 https_cert (tautulli.models.settings.General attribute), 146
 https_cert_chain (tautulli.models.settings.General attribute), 146
 https_create_cert (tautulli.models.settings.General attribute), 147
 https_domain (tautulli.models.settings.General attribute), 147
 https_ip (tautulli.models.settings.General attribute), 147
 https_key (tautulli.models.settings.General attribute), 147
 httpsRequired (tautulli.models.server_list.ServerListEntry attribute), 136
 human_bandwidth (tautulli.models.activity.Session property), 65
 |
 id (tautulli.models.activity.Session attribute), 65
 id (tautulli.models.history.Datum attribute), 81
 id (tautulli.models.metadata.Marker attribute), 94
 id (tautulli.models.metadata.MediaInfoItem attribute), 95
 id (tautulli.models.metadata.Part attribute), 99
 id (tautulli.models.metadata.Stream attribute), 100
 id (tautulli.models.newsletter.Newsletter attribute), 110
 id (tautulli.models.newsletter_config.NewsletterConfig attribute), 106
 id (tautulli.models.newsletter_log.Datum attribute), 108
 id (tautulli.models.notification_log.Datum attribute), 110
 id (tautulli.models.notifiers.Notifier attribute), 112
 id (tautulli.models.search_results.MediaInfoItem attribute), 122
 id (tautulli.models.search_results.Part attribute), 125
 id (tautulli.models.search_results.Stream attribute), 132
 id (tautulli.models.user_ips.Datum attribute), 189
 id_name (tautulli.models.newsletter_config.NewsletterConfig attribute), 106
 identifier (tautulli.models.server_id.ServerID attribute), 134
 IFTTT (class in tautulli.models.settings), 152
 IFTTT (tautulli.models.settings.Settings attribute), 171
 ifttt_enabled (tautulli.models.settings.IFTTT attribute), 152
 ifttt_event (tautulli.models.settings.IFTTT attribute), 152
 ifttt_key (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_buffer (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_concurrent (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_created (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_extdown (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_extup (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_intdown (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_intup (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_newdevice (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_pause (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_play (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_pmsupdate (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_resume (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_stop (tautulli.models.settings.IFTTT attribute), 152
 ifttt_on_watched (tautulli.models.settings.IFTTT attribute), 153
 imgur_client_id (tautulli.models.settings.Monitoring attribute), 154
 import_config (tautulli.models.docs.Docs attribute), 77
 import_config() (tautulli.api.json_api.RawAPI method), 24
 import_config() (tautulli.api.object_api.ObjectAPI method), 54
 import_database (tautulli.models.docs.Docs attribute), 77
 import_database() (tautulli.api.json_api.RawAPI method), 24
 import_database() (tautulli.api.object_api.ObjectAPI method), 54

incl_libraries (tautulli.models.newsletter_config.Config attribute), 103
 indexes (tautulli.models.activity.Session attribute), 65
 indexes (tautulli.models.metadata.Part attribute), 99
 indexes (tautulli.models.search_results.Part attribute), 125
 input_type (tautulli.models.newsletter_config.ConfigOptions attribute), 103
 input_type (tautulli.models.newsletter_config.EmailConfigOptions attribute), 105
 install_type (tautulli.models.update_check.UpdateCheckItem attribute), 183
 interface (tautulli.models.settings.General attribute), 147
 interface_list (tautulli.models.settings.General attribute), 147
 ip (tautulli.models.server_list.ServerListEntry attribute), 136
 ip_address (tautulli.models.activity.Session attribute), 65
 ip_address (tautulli.models.history.Datum attribute), 81
 ip_address (tautulli.models.user_ips.Datum attribute), 189
 ip_address (tautulli.models.user_logins.Datum attribute), 190
 ip_address (tautulli.models.users_table.Datum attribute), 186
 ip_address_public (tautulli.models.activity.Session attribute), 65
 ip_logging_enable (tautulli.models.settings.General attribute), 147
 is_active (tautulli.models.activity.Session attribute), 65
 is_active (tautulli.models.libraries.LibrariesEntry attribute), 87
 is_active (tautulli.models.libraries_table.Datum attribute), 88
 is_active (tautulli.models.library.Library attribute), 86
 is_active (tautulli.models.user.User attribute), 184
 is_active (tautulli.models.users.User attribute), 185
 is_active (tautulli.models.users_table.Datum attribute), 186
 is_admin (tautulli.models.activity.Session attribute), 65
 is_admin (tautulli.models.user.User attribute), 184
 is_admin (tautulli.models.users.User attribute), 185
 is_allow_sync (tautulli.models.activity.Session attribute), 65
 is_allow_sync (tautulli.models.user.User attribute), 184
 is_allow_sync (tautulli.models.users.User attribute), 185
 is_cloud (tautulli.models.server_list.ServerListEntry attribute), 136
 is_home_user (tautulli.models.activity.Session attribute), 65
 is_home_user (tautulli.models.user.User attribute), 184
 is_home_user (tautulli.models.users.User attribute), 185
 is_restricted (tautulli.models.activity.Session attribute), 65
 is_restricted (tautulli.models.user.User attribute), 184
 is_restricted (tautulli.models.users.User attribute), 185
 item_complete_count (tautulli.models.synced_items.SyncedItems attribute), 181
 item_count (tautulli.models.synced_items.SyncedItems attribute), 181
 item_downloaded_count (tautulli.models.synced_items.SyncedItems attribute), 181
 item_downloaded_percent_complete (tautulli.models.synced_items.SyncedItems attribute), 181

J

Join (class in tautulli.models.settings), 153
 Join (tautulli.models.settings.Settings attribute), 171
 join_apikey (tautulli.models.settings.Join attribute), 153
 join_deviceid (tautulli.models.settings.Join attribute), 153
 join_enabled (tautulli.models.settings.Join attribute), 153
 join_incl_subject (tautulli.models.settings.Join attribute), 153
 join_on_buffer (tautulli.models.settings.Join attribute), 153
 join_on_concurrent (tautulli.models.settings.Join attribute), 153
 join_on_created (tautulli.models.settings.Join attribute), 153
 join_on_extdown (tautulli.models.settings.Join attribute), 153
 join_on_extup (tautulli.models.settings.Join attribute), 153
 join_on_intdown (tautulli.models.settings.Join attribute), 153
 join_on_intup (tautulli.models.settings.Join attribute), 153
 join_on_newdevice (tautulli.models.settings.Join attribute), 153
 join_on_pause (tautulli.models.settings.Join attribute), 153
 join_on_play (tautulli.models.settings.Join attribute), 153

- [join_on_pmsupdate](#) ([tautulli.models.settings.Join](#) attribute), 154
[join_on_resume](#) ([tautulli.models.settings.Join](#) attribute), 154
[join_on_stop](#) ([tautulli.models.settings.Join](#) attribute), 154
[join_on_watched](#) ([tautulli.models.settings.Join](#) attribute), 154
[journal_mode](#) ([tautulli.models.settings.Advanced](#) attribute), 138
[jwt_secret](#) ([tautulli.models.settings.Advanced](#) attribute), 138
[jwt_update_secret](#) ([tautulli.models.settings.Advanced](#) attribute), 138
- ## K
- [keep_history](#) ([tautulli.models.activity.Session](#) attribute), 65
[keep_history](#) ([tautulli.models.libraries_table.Datum](#) attribute), 88
[keep_history](#) ([tautulli.models.library.Library](#) attribute), 86
[keep_history](#) ([tautulli.models.user.User](#) attribute), 184
[keep_history](#) ([tautulli.models.users.User](#) attribute), 185
[keep_history](#) ([tautulli.models.users_table.Datum](#) attribute), 187
- ## L
- [label](#) ([tautulli.models.newsletter_config.ConfigOption](#) attribute), 103
[label](#) ([tautulli.models.newsletter_config.EmailConfigOption](#) attribute), 105
[label](#) ([tautulli.models.pms_update.PMSUpdate](#) attribute), 115
[label](#) ([tautulli.models.server_list.ServerListEntry](#) attribute), 136
[labels](#) ([tautulli.models.activity.Session](#) attribute), 65
[labels](#) ([tautulli.models.home_stats.Row](#) attribute), 85
[labels](#) ([tautulli.models.libraries_table.Datum](#) attribute), 88
[labels](#) ([tautulli.models.metadata.Metadata](#) attribute), 97
[labels](#) ([tautulli.models.search_results.EpisodeItem](#) attribute), 119
[labels](#) ([tautulli.models.search_results.MovieItem](#) attribute), 123
[labels](#) ([tautulli.models.search_results.SeasonItem](#) attribute), 127
[labels](#) ([tautulli.models.search_results.ShowItem](#) attribute), 130
[lan_bandwidth](#) ([tautulli.models.activity.Activity](#) attribute), 62
[lan_bandwidth](#) ([tautulli.models.activity.ActivitySummary](#) attribute), 62
[last_accessed](#) ([tautulli.models.libraries_table.Datum](#) attribute), 88
[last_play](#) ([tautulli.models.home_stats.Row](#) attribute), 85
[last_played](#) ([tautulli.models.libraries_table.Datum](#) attribute), 88
[last_played](#) ([tautulli.models.library_media_info.Datum](#) attribute), 91
[last_played](#) ([tautulli.models.user_ips.Datum](#) attribute), 189
[last_played](#) ([tautulli.models.users_table.Datum](#) attribute), 187
[last_refreshed](#) ([tautulli.models.library_media_info.LibraryMediaInfo](#) attribute), 92
[last_seen](#) ([tautulli.models.user_ips.Datum](#) attribute), 189
[last_seen](#) ([tautulli.models.users_table.Datum](#) attribute), 187
[last_viewed_at](#) ([tautulli.models.activity.Session](#) attribute), 65
[last_viewed_at](#) ([tautulli.models.metadata.Metadata](#) attribute), 97
[last_viewed_at](#) ([tautulli.models.search_results.EpisodeItem](#) attribute), 119
[last_viewed_at](#) ([tautulli.models.search_results.MovieItem](#) attribute), 123
[last_viewed_at](#) ([tautulli.models.search_results.SeasonItem](#) attribute), 127
[last_viewed_at](#) ([tautulli.models.search_results.ShowItem](#) attribute), 130
[last_watch](#) ([tautulli.models.home_stats.Row](#) attribute), 85
[latitude](#) ([tautulli.models.geo_ip_lookup.GeoIPLookup](#) attribute), 81
[launch_browser](#) ([tautulli.models.settings.General](#) attribute), 147
[launch_startup](#) ([tautulli.models.settings.General](#) attribute), 147
[leafCount](#) ([tautulli.models.playlists_table.Datum](#) attribute), 113
[level](#) ([tautulli.models.export_fields.MediaInfoField](#) attribute), 80
[level](#) ([tautulli.models.export_fields.MetadataField](#) attribute), 80
[libraries](#) ([tautulli.api.json_api.RawAPI](#) property), 30
[libraries](#) ([tautulli.api.object_api.ObjectAPI](#) property), 60

LibrariesEntry (class in *tautulli.models.libraries*), 87
 LibrariesTable (class in *tautulli.models.libraries_table*), 90
 Library (class in *tautulli.models.library*), 86
 library_art (*tautulli.models.libraries_table.Datum* attribute), 88
 library_art (*tautulli.models.library.Library* attribute), 86
 library_name (*tautulli.models.activity.Session* attribute), 65
 library_name (*tautulli.models.metadata.Metadata* attribute), 97
 library_name (*tautulli.models.search_results.EpisodeItem* attribute), 119
 library_name (*tautulli.models.search_results.MovieItem* attribute), 123
 library_name (*tautulli.models.search_results.SeasonItem* attribute), 127
 library_name (*tautulli.models.search_results.ShowItem* attribute), 130
 library_names (*tautulli.api.json_api.RawAPI* property), 31
 library_names (*tautulli.api.object_api.ObjectAPI* property), 60
 library_thumb (*tautulli.models.libraries_table.Datum* attribute), 88
 library_thumb (*tautulli.models.library.Library* attribute), 86
 LibraryMediaInfo (class in *tautulli.models.library_media_info*), 92
 LibraryName (class in *tautulli.models.library_names*), 92
 librarySectionID (*tautulli.models.collections_table.Datum* attribute), 73
 librarySectionID (*tautulli.models.playlists_table.Datum* attribute), 113
 librarySectionTitle (*tautulli.models.collections_table.Datum* attribute), 73
 LibraryUserStats (class in *tautulli.models.library_user_stats*), 93
 LibraryWatchTimeStats (class in *tautulli.models.library_watch_time_stats*), 93
 live (*tautulli.models.activity.Session* attribute), 65
 live (*tautulli.models.history.Datum* attribute), 81
 live (*tautulli.models.home_stats.Row* attribute), 85
 live (*tautulli.models.libraries_table.Datum* attribute), 89
 live (*tautulli.models.metadata.Metadata* attribute), 97
 live (*tautulli.models.search_results.EpisodeItem* attribute), 119
 live (*tautulli.models.search_results.MovieItem* attribute), 123
 live (*tautulli.models.search_results.SeasonItem* attribute), 127
 live (*tautulli.models.search_results.ShowItem* attribute), 130
 live (*tautulli.models.user_ips.Datum* attribute), 189
 live (*tautulli.models.users_table.Datum* attribute), 187
 live_uuid (*tautulli.models.activity.Session* attribute), 65
 local (*tautulli.models.activity.Session* attribute), 65
 local (*tautulli.models.server_list.ServerListEntry* attribute), 136
 location (*tautulli.models.activity.Session* attribute), 65
 location_milliseconds (*tautulli.models.activity.Session* property), 65
 log_blacklist (*tautulli.models.settings.General* attribute), 147
 log_dir (*tautulli.models.settings.General* attribute), 147
 LogEntry (class in *tautulli.models.logs*), 94
 logging_ignore_interval (*tautulli.models.settings.Monitoring* attribute), 154
 loglevel (*tautulli.models.logs.LogEntry* attribute), 94
 logout_user_session (*tautulli.models.docs.Docs* attribute), 77
 logout_user_session() (*tautulli.api.json_api.RawAPI* method), 25
 logout_user_session() (*tautulli.api.object_api.ObjectAPI* method), 54
 longitude (*tautulli.models.geo_ip_lookup.GeoIPLookup* attribute), 81

M

machine_id (*tautulli.models.activity.Session* attribute), 65
 machine_id (*tautulli.models.history.Datum* attribute), 82
 machine_identifier (*tautulli.models.server_identity.ServerIdentity* attribute), 135
 machine_identifier (*tautulli.models.servers_info.ServersInfoEntry* attribute), 137
 Marker (class in *tautulli.models.metadata*), 94
 markers (*tautulli.models.metadata.Metadata* attribute), 97
 maxmind_license_key (*tautulli.models.settings.General* attribute), 147
 maxYear (*tautulli.models.collections_table.Datum* attribute), 73
 media_index (*tautulli.models.activity.Session* attribute), 65
 media_index (*tautulli.models.history.Datum* attribute), 82

- `media_index` (`tautulli.models.libraries_table.Datum` attribute), 89
- `media_index` (`tautulli.models.library_media_info.Datum` attribute), 91
- `media_index` (`tautulli.models.metadata.Metadata` attribute), 97
- `media_index` (`tautulli.models.search_results.EpisodeItem` attribute), 119
- `media_index` (`tautulli.models.search_results.MovieItem` attribute), 123
- `media_index` (`tautulli.models.search_results.SeasonItem` attribute), 127
- `media_index` (`tautulli.models.search_results.ShowItem` attribute), 130
- `media_index` (`tautulli.models.user_ips.Datum` attribute), 189
- `media_index` (`tautulli.models.users_table.Datum` attribute), 187
- `media_info` (`tautulli.models.metadata.Metadata` attribute), 97
- `media_info` (`tautulli.models.search_results.EpisodeItem` attribute), 119
- `media_info` (`tautulli.models.search_results.MovieItem` attribute), 123
- `media_info` (`tautulli.models.search_results.SeasonItem` attribute), 127
- `media_info` (`tautulli.models.search_results.ShowItem` attribute), 130
- `media_info_fields` (`tautulli.models.export_fields.ExportFields` attribute), 79
- `media_type` (`tautulli.models.activity.Session` attribute), 65
- `media_type` (`tautulli.models.history.Datum` attribute), 82
- `media_type` (`tautulli.models.home_stats.Row` attribute), 85
- `media_type` (`tautulli.models.libraries_table.Datum` attribute), 89
- `media_type` (`tautulli.models.library_media_info.Datum` attribute), 91
- `media_type` (`tautulli.models.metadata.Metadata` attribute), 97
- `media_type` (`tautulli.models.search_results.EpisodeItem` attribute), 119
- `media_type` (`tautulli.models.search_results.MovieItem` attribute), 123
- `media_type` (`tautulli.models.search_results.SeasonItem` attribute), 128
- `media_type` (`tautulli.models.search_results.ShowItem` attribute), 130
- `media_type` (`tautulli.models.stream_data.StreamData` attribute), 178
- `media_type` (`tautulli.models.user_ips.Datum` attribute), 189
- `media_type` (`tautulli.models.users_table.Datum` attribute), 187
- `MediaInfoField` (class in `tautulli.models.export_fields`), 80
- `MediaInfoItem` (class in `tautulli.models.metadata`), 95
- `MediaInfoItem` (class in `tautulli.models.search_results`), 121
- `message` (`tautulli.models.activity.ActivitySummary` property), 63
- `Metadata` (class in `tautulli.models.metadata`), 96
- `metadata_cache_seconds` (`tautulli.models.settings.Advanced` attribute), 138
- `metadata_fields` (`tautulli.models.export_fields.ExportFields` attribute), 79
- `metadata_type` (`tautulli.models.synced_items.SyncedItems` attribute), 181
- `MetadataField` (class in `tautulli.models.export_fields`), 80
- `minYear` (`tautulli.models.collections_table.Datum` attribute), 73
- `model_config` (`tautulli.models.activity.Activity` attribute), 62
- `model_config` (`tautulli.models.activity.ActivitySummary` attribute), 63
- `model_config` (`tautulli.models.activity.Session` attribute), 65
- `model_config` (`tautulli.models.collections_table.CollectionsTable` attribute), 72
- `model_config` (`tautulli.models.collections_table.Datum` attribute), 73
- `model_config` (`tautulli.models.date_formats.DateFormats` attribute), 74
- `model_config` (`tautulli.models.docs.Docs` attribute), 77
- `model_config` (`tautulli.models.export_fields.ExportFields` attribute), 79
- `model_config` (`tautulli.models.export_fields.MediaInfoField` attribute), 80
- `model_config` (`tautulli.models.export_fields.MetadataField` attribute), 80
- `model_config` (`tautulli.models.geo_ip_lookup.GeoIPLookup` attribute), 81
- `model_config` (`tautulli.models.history.Datum` attribute), 82
- `model_config` (`tautulli.models.history.History` attribute), 84
- `model_config` (`tautulli.models.home_stats.HomeStat` attribute), 84
- `model_config` (`tautulli.models.home_stats.Row` attribute), 85
- `model_config` (`tautulli.models.libraries.LibrariesEntry` attribute), 87

`model_config` (`tautulli.models.libraries_table.Datum` attribute), 89
`model_config` (`tautulli.models.libraries_table.LibrariesTable` attribute), 90
`model_config` (`tautulli.models.library.Library` attribute), 86
`model_config` (`tautulli.models.library_media_info.Datum` attribute), 91
`model_config` (`tautulli.models.library_media_info.LibraryMediaInfo` attribute), 92
`model_config` (`tautulli.models.library_names.LibraryNames` attribute), 92
`model_config` (`tautulli.models.library_user_stats.LibraryUserStats` attribute), 93
`model_config` (`tautulli.models.library_watch_time_stats.LibraryWatchTimeStats` attribute), 93
`model_config` (`tautulli.models.logs.LogEntry` attribute), 94
`model_config` (`tautulli.models.metadata.Marker` attribute), 94
`model_config` (`tautulli.models.metadata.MediaInfoItem` attribute), 95
`model_config` (`tautulli.models.metadata.Metadata` attribute), 97
`model_config` (`tautulli.models.metadata.Part` attribute), 99
`model_config` (`tautulli.models.metadata.Stream` attribute), 100
`model_config` (`tautulli.models.new_rating_keys.Field0` attribute), 102
`model_config` (`tautulli.models.new_rating_keys.NewRatingKeys` attribute), 102
`model_config` (`tautulli.models.newsletter.Newsletter` attribute), 110
`model_config` (`tautulli.models.newsletter_config.Config` attribute), 103
`model_config` (`tautulli.models.newsletter_config.ConfigOptions` attribute), 103
`model_config` (`tautulli.models.newsletter_config.EmailConfig` attribute), 104
`model_config` (`tautulli.models.newsletter_config.EmailConfigOptions` attribute), 105
`model_config` (`tautulli.models.newsletter_config.MovieLibrary` attribute), 105
`model_config` (`tautulli.models.newsletter_config.MusicLibrary` attribute), 105
`model_config` (`tautulli.models.newsletter_config.NewsletterConfig` attribute), 106
`model_config` (`tautulli.models.newsletter_config.OtherVideoLibrary` attribute), 107
`model_config` (`tautulli.models.newsletter_config.SelectOptions` attribute), 107
`model_config` (`tautulli.models.newsletter_config.SelectOptions` attribute), 107
`model_config` (`tautulli.models.newsletter_config.TVShowLibrary` attribute), 108
`model_config` (`tautulli.models.newsletter_log.Datum` attribute), 108
`model_config` (`tautulli.models.newsletter_log.NewsletterLog` attribute), 109
`model_config` (`tautulli.models.notification_log.Datum` attribute), 110
`model_config` (`tautulli.models.notification_log.NotificationLog` attribute), 111
`model_config` (`tautulli.models.notifier_parameters.NotifierParameter` attribute), 112
`model_config` (`tautulli.models.notifiers.Notifier` attribute), 112
`model_config` (`tautulli.models.old_rating_keys.Field0` attribute), 113
`model_config` (`tautulli.models.old_rating_keys.OldRatingKeys` attribute), 113
`model_config` (`tautulli.models.playlists_table.Datum` attribute), 113
`model_config` (`tautulli.models.playlists_table.PlaylistsTable` attribute), 114
`model_config` (`tautulli.models.plex_log.PlexLog` attribute), 115
`model_config` (`tautulli.models.pms_update.PMSUpdate` attribute), 115
`model_config` (`tautulli.models.recently_added.RecentlyAdded` attribute), 116
`model_config` (`tautulli.models.registered_device.RegisteredDevice` attribute), 116
`model_config` (`tautulli.models.search_results.EpisodeItem` attribute), 119
`model_config` (`tautulli.models.search_results.MediaInfoItem` attribute), 122
`model_config` (`tautulli.models.search_results.MovieItem` attribute), 123
`model_config` (`tautulli.models.search_results.Part` attribute), 125
`model_config` (`tautulli.models.search_results.ResultsList` attribute), 126
`model_config` (`tautulli.models.search_results.SearchResults` attribute), 126
`model_config` (`tautulli.models.search_results.SeasonItem` attribute), 128
`model_config` (`tautulli.models.search_results.ShowItem` attribute), 130
`model_config` (`tautulli.models.search_results.Stream` attribute), 132
`model_config` (`tautulli.models.server_id.ServerID` attribute), 134
`model_config` (`tautulli.models.server_identity.ServerIdentity` attribute), 135
`model_config` (`tautulli.models.server_info.ServerInfo` attribute), 135

`model_config(tautulli.models.server_list.ServerListEntry attribute), 136`
`model_config(tautulli.models.server_status.ServerStatus attribute), 137`
`model_config(tautulli.models.servers_info.ServersInfoEntry attribute), 137`
`model_config(tautulli.models.settings.Advanced attribute), 138`
`model_config(tautulli.models.settings.Boxcar attribute), 139`
`model_config(tautulli.models.settings.Browser attribute), 141`
`model_config(tautulli.models.settings.Cloudinary attribute), 141`
`model_config(tautulli.models.settings.Email attribute), 142`
`model_config(tautulli.models.settings.Facebook attribute), 144`
`model_config(tautulli.models.settings.General attribute), 147`
`model_config(tautulli.models.settings.GetFilesizesHold attribute), 149`
`model_config(tautulli.models.settings.Growl attribute), 150`
`model_config(tautulli.models.settings.Hipchat attribute), 151`
`model_config(tautulli.models.settings.IFTTT attribute), 153`
`model_config(tautulli.models.settings.Join attribute), 154`
`model_config(tautulli.models.settings.Monitoring attribute), 154`
`model_config(tautulli.models.settings.Newsletter attribute), 159`
`model_config(tautulli.models.settings.NMA attribute), 158`
`model_config(tautulli.models.settings.OSXNotify attribute), 160`
`model_config(tautulli.models.settings.Plex attribute), 162`
`model_config(tautulli.models.settings.PlexWatch attribute), 163`
`model_config(tautulli.models.settings.PMS attribute), 161`
`model_config(tautulli.models.settings.Prowl attribute), 164`
`model_config(tautulli.models.settings.Pushalot attribute), 166`
`model_config(tautulli.models.settings.PushBullet attribute), 165`
`model_config(tautulli.models.settings.Pushover attribute), 167`
`model_config(tautulli.models.settings.Scripts attribute), 168`
`model_config(tautulli.models.settings.Settings attribute), 171`
`model_config(tautulli.models.settings.Slack attribute), 172`
`model_config(tautulli.models.settings.Telegram attribute), 174`
`model_config(tautulli.models.settings.Twitter attribute), 175`
`model_config(tautulli.models.settings.XBMC attribute), 177`
`model_config(tautulli.models.stream_data.StreamData attribute), 178`
`model_config(tautulli.models.synced_items.SyncedItems attribute), 181`
`model_config(tautulli.models.tautulli_info.TautulliInfo attribute), 183`
`model_config(tautulli.models.update_check.UpdateCheck attribute), 183`
`model_config(tautulli.models.user.User attribute), 184`
`model_config(tautulli.models.user_ips.Datum attribute), 189`
`model_config(tautulli.models.user_ips.UserIPs attribute), 190`
`model_config(tautulli.models.user_logins.Datum attribute), 190`
`model_config(tautulli.models.user_logins.UserLogins attribute), 191`
`model_config(tautulli.models.user_player_stats.UserPlayerStats attribute), 192`
`model_config(tautulli.models.user_watch_time_stats.UserWatchTimeStats attribute), 193`
`model_config(tautulli.models.usernames.UserName attribute), 192`
`model_config(tautulli.models.users.User attribute), 185`
`model_config(tautulli.models.users_table.Datum attribute), 187`
`model_config(tautulli.models.users_table.UsersTable attribute), 188`
`model_config(tautulli.models.whois_lookup.Net attribute), 193`
`model_config(tautulli.models.whois_lookup.WHOISLookup attribute), 194`
`model_fields(tautulli.models.activity.Activity attribute), 62`
`model_fields(tautulli.models.activity.ActivitySummary attribute), 63`
`model_fields(tautulli.models.activity.Session attribute), 65`
`model_fields(tautulli.models.collections_table.CollectionsTable attribute), 72`
`model_fields(tautulli.models.collections_table.Datum attribute), 73`
`model_fields(tautulli.models.date_formats.DateFormats attribute), 73`

attribute), 74
 model_fields (tautulli.models.docs.Docs attribute), 77
 model_fields (tautulli.models.export_fields.ExportFields attribute), 79
 model_fields (tautulli.models.export_fields.MediaInfoField attribute), 80
 model_fields (tautulli.models.export_fields.MetadataField attribute), 80
 model_fields (tautulli.models.geo_ip_lookup.GeoIPLookup attribute), 81
 model_fields (tautulli.models.history.Datum attribute), 82
 model_fields (tautulli.models.history.History attribute), 84
 model_fields (tautulli.models.home_stats.HomeStat attribute), 84
 model_fields (tautulli.models.home_stats.Row attribute), 85
 model_fields (tautulli.models.libraries.LibrariesEntry attribute), 87
 model_fields (tautulli.models.libraries_table.Datum attribute), 89
 model_fields (tautulli.models.libraries_table.LibrariesTable attribute), 90
 model_fields (tautulli.models.library.Library attribute), 86
 model_fields (tautulli.models.library_media_info.Datum attribute), 91
 model_fields (tautulli.models.library_media_info.LibraryMediaInfo attribute), 92
 model_fields (tautulli.models.library_names.LibraryNames attribute), 92
 model_fields (tautulli.models.library_user_stats.LibraryUserStats attribute), 93
 model_fields (tautulli.models.library_watch_time_stats.LibraryWatchTimeStats attribute), 93
 model_fields (tautulli.models.logs.LogEntry attribute), 94
 model_fields (tautulli.models.metadata.Marker attribute), 94
 model_fields (tautulli.models.metadata.MediaInfoItem attribute), 95
 model_fields (tautulli.models.metadata.Metadata attribute), 97
 model_fields (tautulli.models.metadata.Part attribute), 100
 model_fields (tautulli.models.metadata.Stream attribute), 100
 model_fields (tautulli.models.new_rating_keys.Field0 attribute), 102
 model_fields (tautulli.models.new_rating_keys.NewRatingKeys attribute), 102
 model_fields (tautulli.models.newsletter.Newsletter attribute), 110
 model_fields (tautulli.models.newsletter_config.Config attribute), 103
 model_fields (tautulli.models.newsletter_config.ConfigOption attribute), 103
 model_fields (tautulli.models.newsletter_config.EmailConfig attribute), 104
 model_fields (tautulli.models.newsletter_config.EmailConfigOption attribute), 105
 model_fields (tautulli.models.newsletter_config.MovieLibrary attribute), 105
 model_fields (tautulli.models.newsletter_config.MusicLibrary attribute), 105
 model_fields (tautulli.models.newsletter_config.NewsletterConfig attribute), 106
 model_fields (tautulli.models.newsletter_config.OtherVideoLibrary attribute), 107
 model_fields (tautulli.models.newsletter_config.SelectOption attribute), 107
 model_fields (tautulli.models.newsletter_config.SelectOptions attribute), 107
 model_fields (tautulli.models.newsletter_config.TVShowLibrary attribute), 108
 model_fields (tautulli.models.newsletter_log.Datum attribute), 108
 model_fields (tautulli.models.newsletter_log.NewsletterLog attribute), 109
 model_fields (tautulli.models.notification_log.Datum attribute), 110
 model_fields (tautulli.models.notification_log.NotificationLog attribute), 111
 model_fields (tautulli.models.notifier_parameters.NotifierParameter attribute), 112
 model_fields (tautulli.models.notifiers.Notifier attribute), 112
 model_fields (tautulli.models.old_rating_keys.Field0 attribute), 113
 model_fields (tautulli.models.old_rating_keys.OldRatingKeys attribute), 113
 model_fields (tautulli.models.playlists_table.Datum attribute), 113
 model_fields (tautulli.models.playlists_table.PlaylistsTable attribute), 114
 model_fields (tautulli.models.plex_log.PlexLog attribute), 115
 model_fields (tautulli.models.pms_update.PMSUpdate attribute), 115
 model_fields (tautulli.models.recently_added.RecentlyAdded attribute), 116
 model_fields (tautulli.models.registered_device.RegisteredDevice attribute), 116
 model_fields (tautulli.models.search_results.EpisodeItem attribute), 119
 model_fields (tautulli.models.search_results.MediaInfoItem attribute), 122

<code>model_fields(tautulli.models.search_results.MovieItem attribute), 123</code>	<code>model_fields(tautulli.models.settings.NMA attribute), 158</code>
<code>model_fields(tautulli.models.search_results.Part attribute), 125</code>	<code>model_fields(tautulli.models.settings.OSXNotify attribute), 160</code>
<code>model_fields(tautulli.models.search_results.ResultsList attribute), 126</code>	<code>model_fields(tautulli.models.settings.Plex attribute), 162</code>
<code>model_fields(tautulli.models.search_results.SearchResult attribute), 126</code>	<code>model_fields(tautulli.models.settings.PlexWatch attribute), 163</code>
<code>model_fields(tautulli.models.search_results.SeasonItem attribute), 128</code>	<code>model_fields(tautulli.models.settings.PMS attribute), 161</code>
<code>model_fields(tautulli.models.search_results.ShowItem attribute), 130</code>	<code>model_fields(tautulli.models.settings.Prowl attribute), 164</code>
<code>model_fields(tautulli.models.search_results.Stream attribute), 133</code>	<code>model_fields(tautulli.models.settings.Pushalot attribute), 166</code>
<code>model_fields(tautulli.models.server_id.ServerID attribute), 134</code>	<code>model_fields(tautulli.models.settings.PushBullet attribute), 165</code>
<code>model_fields(tautulli.models.server_identity.ServerIdentity attribute), 135</code>	<code>model_fields(tautulli.models.settings.Pushover attribute), 167</code>
<code>model_fields(tautulli.models.server_info.ServerInfo attribute), 135</code>	<code>model_fields(tautulli.models.settings.Scripts attribute), 168</code>
<code>model_fields(tautulli.models.server_list.ServerListEntry attribute), 136</code>	<code>model_fields(tautulli.models.settings.Settings attribute), 171</code>
<code>model_fields(tautulli.models.server_status.ServerStatus attribute), 137</code>	<code>model_fields(tautulli.models.settings.Slack attribute), 172</code>
<code>model_fields(tautulli.models.servers_info.ServersInfoEntry attribute), 137</code>	<code>model_fields(tautulli.models.settings.Telegram attribute), 174</code>
<code>model_fields(tautulli.models.settings.Advanced attribute), 138</code>	<code>model_fields(tautulli.models.settings.Twitter attribute), 175</code>
<code>model_fields(tautulli.models.settings.Boxcar attribute), 140</code>	<code>model_fields(tautulli.models.settings.XBMC attribute), 177</code>
<code>model_fields(tautulli.models.settings.Browser attribute), 141</code>	<code>model_fields(tautulli.models.stream_data.StreamData attribute), 178</code>
<code>model_fields(tautulli.models.settings.Cloudinary attribute), 141</code>	<code>model_fields(tautulli.models.synced_items.SyncedItems attribute), 181</code>
<code>model_fields(tautulli.models.settings.Email attribute), 142</code>	<code>model_fields(tautulli.models.tautulli_info.TautulliInfo attribute), 183</code>
<code>model_fields(tautulli.models.settings.Facebook attribute), 144</code>	<code>model_fields(tautulli.models.update_check.UpdateCheck attribute), 183</code>
<code>model_fields(tautulli.models.settings.General attribute), 147</code>	<code>model_fields(tautulli.models.user.User attribute), 184</code>
<code>model_fields(tautulli.models.settings.GetFilesSizesHold attribute), 149</code>	<code>model_fields(tautulli.models.user_ips.Datum attribute), 189</code>
<code>model_fields(tautulli.models.settings.Growl attribute), 150</code>	<code>model_fields(tautulli.models.user_ips.UserIPs attribute), 190</code>
<code>model_fields(tautulli.models.settings.Hipchat attribute), 151</code>	<code>model_fields(tautulli.models.user_logins.Datum attribute), 191</code>
<code>model_fields(tautulli.models.settings.IFTTT attribute), 153</code>	<code>model_fields(tautulli.models.user_logins.UserLogins attribute), 191</code>
<code>model_fields(tautulli.models.settings.Join attribute), 154</code>	<code>model_fields(tautulli.models.user_player_stats.UserPlayerStats attribute), 192</code>
<code>model_fields(tautulli.models.settings.Monitoring attribute), 154</code>	<code>model_fields(tautulli.models.user_watch_time_stats.UserWatchTimeStats attribute), 193</code>
<code>model_fields(tautulli.models.settings.Newsletter attribute), 159</code>	<code>model_fields(tautulli.models.usernames.UserName attribute), 192</code>
	<code>model_fields(tautulli.models.users.User attribute),</code>

- 185
- `model_fields` (*tautulli.models.users_table.Datum* attribute), 187
- `model_fields` (*tautulli.models.users_table.UsersTable* attribute), 188
- `model_fields` (*tautulli.models.whois_lookup.Net* attribute), 193
- `model_fields` (*tautulli.models.whois_lookup.WHOISLookup* attribute), 194
- module
- `tautulli.models.activity`, 62
 - `tautulli.models.collections_table`, 72
 - `tautulli.models.date_formats`, 74
 - `tautulli.models.docs`, 74
 - `tautulli.models.export_fields`, 79
 - `tautulli.models.geo_ip_lookup`, 80
 - `tautulli.models.history`, 81
 - `tautulli.models.home_stats`, 84
 - `tautulli.models.libraries`, 87
 - `tautulli.models.libraries_table`, 88
 - `tautulli.models.library`, 86
 - `tautulli.models.library_media_info`, 90
 - `tautulli.models.library_names`, 92
 - `tautulli.models.library_user_stats`, 93
 - `tautulli.models.library_watch_time_stats`, 93
 - `tautulli.models.logs`, 94
 - `tautulli.models.metadata`, 94
 - `tautulli.models.new_rating_keys`, 102
 - `tautulli.models.newsletter`, 110
 - `tautulli.models.newsletter_config`, 103
 - `tautulli.models.newsletter_log`, 108
 - `tautulli.models.notification_log`, 110
 - `tautulli.models.notifier_parameters`, 112
 - `tautulli.models.notifiers`, 112
 - `tautulli.models.old_rating_keys`, 113
 - `tautulli.models.playlists_table`, 113
 - `tautulli.models.plex_log`, 115
 - `tautulli.models.pms_update`, 115
 - `tautulli.models.recently_added`, 116
 - `tautulli.models.registered_device`, 116
 - `tautulli.models.search_results`, 118
 - `tautulli.models.server_id`, 134
 - `tautulli.models.server_identity`, 135
 - `tautulli.models.server_info`, 135
 - `tautulli.models.server_list`, 136
 - `tautulli.models.server_status`, 137
 - `tautulli.models.servers_info`, 137
 - `tautulli.models.settings`, 138
 - `tautulli.models.stream_data`, 178
 - `tautulli.models.synced_items`, 181
 - `tautulli.models.tautulli_info`, 183
 - `tautulli.models.update_check`, 183
 - `tautulli.models.user`, 184
 - `tautulli.models.user_ips`, 189
 - `tautulli.models.user_logins`, 190
 - `tautulli.models.user_player_stats`, 192
 - `tautulli.models.user_watch_time_stats`, 193
 - `tautulli.models.usernames`, 192
 - `tautulli.models.users`, 185
 - `tautulli.models.users_table`, 186
 - `tautulli.models.whois_lookup`, 193
- `monitor_pms_updates` (*tautulli.models.settings.Monitoring* attribute), 156
- `monitor_remote_access` (*tautulli.models.settings.Monitoring* attribute), 156
- `Monitoring` (class in *tautulli.models.settings*), 154
- `Monitoring` (*tautulli.models.settings.Settings* attribute), 171
- `monitoring_interval` (*tautulli.models.settings.Monitoring* attribute), 156
- `monitoring_use_websocket` (*tautulli.models.settings.Monitoring* attribute), 156
- `movie` (*tautulli.models.search_results.ResultsList* attribute), 126
- `MovieLibraries` (*tautulli.models.newsletter_config.SelectOptions* attribute), 107
- `movie_logging_enable` (*tautulli.models.settings.Monitoring* attribute), 156
- `movie_notify_enable` (*tautulli.models.settings.Monitoring* attribute), 156
- `movie_notify_on_pause` (*tautulli.models.settings.Monitoring* attribute), 156
- `movie_notify_on_start` (*tautulli.models.settings.Monitoring* attribute), 156
- `movie_notify_on_stop` (*tautulli.models.settings.Monitoring* attribute), 156
- `movie_watched_percent` (*tautulli.models.settings.Monitoring* attribute), 156
- `MovieItem` (class in *tautulli.models.search_results*), 122
- `MovieLibrary` (class in *tautulli.models.newsletter_config*), 105
- `msg` (*tautulli.models.logs.LogEntry* attribute), 94
- `MusicLibraries` (*tautulli.models.newsletter_config.SelectOptions* attribute), 107

music_logging_enable	(tautulli.models.settings.Monitoring attribute), 156	newsletter_password	(tautulli.models.settings.Newsletter attribute), 159
music_notify_enable	(tautulli.models.settings.Monitoring attribute), 156	newsletter_self_hosted	(tautulli.models.settings.Newsletter attribute), 160
music_notify_on_pause	(tautulli.models.settings.Monitoring attribute), 156	newsletter_static_url	(tautulli.models.settings.Newsletter attribute), 160
music_notify_on_start	(tautulli.models.settings.Monitoring attribute), 156	newsletter_templates	(tautulli.models.settings.Newsletter attribute), 160
music_notify_on_stop	(tautulli.models.settings.Monitoring attribute), 156	NewsletterConfig	(class in tautulli.models.newsletter_config), 106
music_watched_percent	(tautulli.models.settings.Monitoring attribute), 156	NewsletterLog	(class in tautulli.models.newsletter_log), 109
musicbrainz_lookup	(tautulli.models.settings.General attribute), 149	newsletters	(tautulli.api.json_api.RawAPI property), 31
MusicLibrary	(class in tautulli.models.newsletter_config), 105	newsletters	(tautulli.api.object_api.ObjectAPI property), 60
N		NMA	(class in tautulli.models.settings), 158
name	(tautulli.models.newsletter_config.ConfigOption attribute), 104	NMA	(tautulli.models.settings.Settings attribute), 171
name	(tautulli.models.newsletter_config.EmailConfigOption attribute), 105	nma_apikey	(tautulli.models.settings.NMA attribute), 158
name	(tautulli.models.notifier_parameters.NotifierParameter attribute), 112	nma_enabled	(tautulli.models.settings.NMA attribute), 158
name	(tautulli.models.servers_info.ServersInfoEntry attribute), 137	nma_on_buffer	(tautulli.models.settings.NMA attribute), 158
name	(tautulli.models.whois_lookup.Net attribute), 194	nma_on_concurrent	(tautulli.models.settings.NMA attribute), 159
Net	(class in tautulli.models.whois_lookup), 193	nma_on_created	(tautulli.models.settings.NMA attribute), 159
nets	(tautulli.models.whois_lookup.WHOISLookup attribute), 194	nma_on_extdown	(tautulli.models.settings.NMA attribute), 159
NewRatingKeys	(class in tautulli.models.new_rating_keys), 102	nma_on_extup	(tautulli.models.settings.NMA attribute), 159
Newsletter	(class in tautulli.models.newsletter), 110	nma_on_intdown	(tautulli.models.settings.NMA attribute), 159
Newsletter	(class in tautulli.models.settings), 159	nma_on_intup	(tautulli.models.settings.NMA attribute), 159
Newsletter	(tautulli.models.settings.Settings attribute), 171	nma_on_newdevice	(tautulli.models.settings.NMA attribute), 159
newsletter_auth	(tautulli.models.settings.Newsletter attribute), 159	nma_on_pause	(tautulli.models.settings.NMA attribute), 159
newsletter_custom_dir	(tautulli.models.settings.Newsletter attribute), 159	nma_on_play	(tautulli.models.settings.NMA attribute), 159
newsletter_dir	(tautulli.models.settings.Newsletter attribute), 159	nma_on_pmsupdate	(tautulli.models.settings.NMA attribute), 159
newsletter_id	(tautulli.models.newsletter_log.Datum attribute), 109	nma_on_resume	(tautulli.models.settings.NMA attribute), 159
newsletter_inline_styles	(tautulli.models.settings.Newsletter attribute), 159	nma_on_stop	(tautulli.models.settings.NMA attribute), 159
		nma_on_watched	(tautulli.models.settings.NMA attribute), 159

<code>nma_priority</code> (<i>tautulli.models.settings.NMA</i> attribute), 159	<code>method</code>), 25	
<code>notification_threads</code> (<i>tautulli.models.settings.Advanced</i> attribute), 138	<code>notify_newsletter()</code> (<i>tautulli.api.object_api.ObjectAPI</i> method), 55	
<code>NotificationLog</code> (class in <i>tautulli.models.notification_log</i>), 111	<code>notify_on_buffer_body_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156	
<code>Notifier</code> (class in <i>tautulli.models.notifiers</i>), 112	<code>notify_on_buffer_subject_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156	
<code>notifier_id</code> (<i>tautulli.models.newsletter_config.Config</i> attribute), 103	<code>notify_on_concurrent_body_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156	
<code>notifier_id</code> (<i>tautulli.models.newsletter_config.EmailConfig</i> attribute), 104	<code>notify_on_concurrent_subject_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156	
<code>notifier_id</code> (<i>tautulli.models.notification_log.Datum</i> attribute), 111	<code>notify_on_created_body_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156	
<code>notifier_parameters</code> (<i>tautulli.api.json_api.RawAPI</i> property), 31	<code>notify_on_created_subject_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 157	
<code>notifier_parameters</code> (<i>tautulli.api.object_api.ObjectAPI</i> property), 60	<code>notify_on_extdown_body_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 157	
<code>NotifierParameter</code> (class in <i>tautulli.models.notifier_parameters</i>), 112	<code>notify_on_extdown_subject_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 157	
<code>notify</code> (<i>tautulli.models.docs.Docs</i> attribute), 79	<code>notify_on_extup_body_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 157	
<code>notify()</code> (<i>tautulli.api.json_api.RawAPI</i> method), 25	<code>notify_on_extup_subject_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 157	
<code>notify()</code> (<i>tautulli.api.object_api.ObjectAPI</i> method), 55	<code>notify_on_intdown_body_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 157	
<code>notify_action</code> (<i>tautulli.models.newsletter_log.Datum</i> attribute), 109	<code>notify_on_intdown_subject_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 157	
<code>notify_action</code> (<i>tautulli.models.notification_log.Datum</i> attribute), 111	<code>notify_on_intup_body_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 157	
<code>notify_concurrent_by_ip</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156	<code>notify_on_intup_subject_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 157	
<code>notify_concurrent_threshold</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156	<code>notify_on_newdevice_body_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 157	
<code>notify_consecutive</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156	<code>notify_on_newdevice_subject_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 157	
<code>notify_continued_session_threshold</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156	<code>notify_on_pause_body_text</code> (<i>tautulli.models.settings.Monitoring</i> attribute),	
<code>notify_group_recently_added</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156		
<code>notify_group_recently_added_grandparent</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156		
<code>notify_group_recently_added_parent</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156		
<code>notify_new_device_initial_only</code> (<i>tautulli.models.settings.Monitoring</i> attribute), 156		
<code>notify_newsletter</code> (<i>tautulli.models.docs.Docs</i> attribute), 79		
<code>notify_newsletter()</code> (<i>tautulli.api.json_api.RawAPI</i>		

157			
notify_on_pause_subject_text	(tautulli.models.settings.Monitoring attribute), 157	notify_scripts_args_text	(tautulli.models.settings.Monitoring attribute), 157
notify_on_pmsupdate_body_text	(tautulli.models.settings.Monitoring attribute), 157	notify_upload_posters	(tautulli.models.settings.Monitoring attribute), 157
notify_on_pmsupdate_subject_text	(tautulli.models.settings.Monitoring attribute), 157	notify_watched_percent	(tautulli.models.settings.Monitoring attribute), 157
notify_on_resume_body_text	(tautulli.models.settings.Monitoring attribute), 157	O	
notify_on_resume_subject_text	(tautulli.models.settings.Monitoring attribute), 157	ObjectAPI	(class in <i>tautulli.api.object_api</i>), 33
notify_on_start_body_text	(tautulli.models.settings.Monitoring attribute), 157	OldRatingKeys	(class in <i>tautulli.models.old_rating_keys</i>), 113
notify_on_start_subject_text	(tautulli.models.settings.Monitoring attribute), 157	optimized_version	(<i>tautulli.models.activity.Session</i> attribute), 67
notify_on_stop_body_text	(tautulli.models.settings.Monitoring attribute), 157	optimized_version	(<i>tautulli.models.metadata.MediaInfoItem</i> attribute), 96
notify_on_stop_subject_text	(tautulli.models.settings.Monitoring attribute), 157	optimized_version	(<i>tautulli.models.search_results.MediaInfoItem</i> attribute), 122
notify_on_watched_body_text	(tautulli.models.settings.Monitoring attribute), 157	optimized_version	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180
notify_on_watched_subject_text	(tautulli.models.settings.Monitoring attribute), 157	optimized_version_profile	(<i>tautulli.models.activity.Session</i> attribute), 67
notify_recently_added	(<i>tautulli.models.docs.Docs</i> attribute), 79	optimized_version_profile	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180
notify_recently_added	(tautulli.models.settings.Monitoring attribute), 157	optimized_version_title	(<i>tautulli.models.activity.Session</i> attribute), 67
notify_recently_added()	(<i>tautulli.api.json_api.RawAPI</i> method), 25	optimized_version_title	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180
notify_recently_added()	(<i>tautulli.api.object_api.ObjectAPI</i> method), 55	original_title	(<i>tautulli.models.activity.Session</i> attribute), 67
notify_recently_added_delay	(tautulli.models.settings.Monitoring attribute), 157	original_title	(<i>tautulli.models.history.Datum</i> attribute), 83
notify_recently_added_grandparent	(tautulli.models.settings.Monitoring attribute), 157	original_title	(<i>tautulli.models.metadata.Metadata</i> attribute), 99
notify_recently_added_upgrade	(tautulli.models.settings.Monitoring attribute), 157	original_title	(<i>tautulli.models.search_results.EpisodeItem</i> attribute), 120
notify_remote_access_threshold	(tautulli.models.settings.Monitoring attribute), 157	original_title	(<i>tautulli.models.search_results.MovieItem</i> attribute), 124
		original_title	(<i>tautulli.models.search_results.SeasonItem</i> attribute), 129
		original_title	(<i>tautulli.models.search_results.ShowItem</i> attribute), 129

- tribute), 131
- original_title (tautulli.models.stream_data.StreamData attribute), 180
- originally_available_at (tautulli.models.activity.Session attribute), 67
- originally_available_at (tautulli.models.history.Datum attribute), 83
- originally_available_at (tautulli.models.libraries_table.Datum attribute), 89
- originally_available_at (tautulli.models.metadata.Metadata attribute), 99
- originally_available_at (tautulli.models.search_results.EpisodeItem attribute), 120
- originally_available_at (tautulli.models.search_results.MovieItem attribute), 124
- originally_available_at (tautulli.models.search_results.SeasonItem attribute), 129
- originally_available_at (tautulli.models.search_results.ShowItem attribute), 131
- originally_available_at (tautulli.models.user_ips.Datum attribute), 189
- originally_available_at (tautulli.models.users_table.Datum attribute), 187
- os (tautulli.models.user_logins.Datum attribute), 191
- OSX_Notify (tautulli.models.settings.Settings attribute), 171
- osx_notify_app (tautulli.models.settings.OSXNotify attribute), 160
- osx_notify_enabled (tautulli.models.settings.OSXNotify attribute), 160
- osx_notify_on_buffer (tautulli.models.settings.OSXNotify attribute), 160
- osx_notify_on_concurrent (tautulli.models.settings.OSXNotify attribute), 160
- osx_notify_on_created (tautulli.models.settings.OSXNotify attribute), 160
- osx_notify_on_extdown (tautulli.models.settings.OSXNotify attribute), 160
- osx_notify_on_extup (tautulli.models.settings.OSXNotify attribute), 160
- osx_notify_on_intdown (tautulli.models.settings.OSXNotify attribute), 160
- osx_notify_on_intup (tautulli.models.settings.OSXNotify attribute), 160
- osx_notify_on_newdevice (tautulli.models.settings.OSXNotify attribute), 160
- osx_notify_on_pause (tautulli.models.settings.OSXNotify attribute), 160
- osx_notify_on_play (tautulli.models.settings.OSXNotify attribute), 161
- osx_notify_on_pmsupdate (tautulli.models.settings.OSXNotify attribute), 161
- osx_notify_on_resume (tautulli.models.settings.OSXNotify attribute), 161
- osx_notify_on_stop (tautulli.models.settings.OSXNotify attribute), 161
- osx_notify_on_watched (tautulli.models.settings.OSXNotify attribute), 161
- OSXNotify (class in tautulli.models.settings), 160
- Other_Video_Libraries (tautulli.models.newsletter_config.SelectOptions attribute), 107
- OtherVideoLibrary (class in tautulli.models.newsletter_config), 107
- ## P
- parent_count (tautulli.models.libraries.LibrariesEntry attribute), 88
- parent_count (tautulli.models.libraries_table.Datum attribute), 89
- parent_count (tautulli.models.library.Library attribute), 87
- parent_guid (tautulli.models.activity.Session attribute), 67
- parent_guid (tautulli.models.metadata.Metadata attribute), 99
- parent_guid (tautulli.models.search_results.EpisodeItem attribute), 121
- parent_guid (tautulli.models.search_results.MovieItem attribute), 125
- parent_guid (tautulli.models.search_results.SeasonItem attribute), 129
- parent_guid (tautulli.models.search_results.ShowItem attribute), 132

parent_guids (*tautulli.models.metadata.Metadata* attribute), 99
parent_media_index (*tautulli.models.activity.Session* attribute), 67
parent_media_index (*tautulli.models.history.Datum* attribute), 83
parent_media_index (*tautulli.models.libraries_table.Datum* attribute), 89
parent_media_index (*tautulli.models.library_media_info.Datum* attribute), 91
parent_media_index (*tautulli.models.metadata.Metadata* attribute), 99
parent_media_index (*tautulli.models.search_results.EpisodeItem* attribute), 121
parent_media_index (*tautulli.models.search_results.MovieItem* attribute), 125
parent_media_index (*tautulli.models.search_results.SeasonItem* attribute), 129
parent_media_index (*tautulli.models.search_results.ShowItem* attribute), 132
parent_media_index (*tautulli.models.user_ips.Datum* attribute), 189
parent_media_index (*tautulli.models.users_table.Datum* attribute), 187
parent_rating_key (*tautulli.models.activity.Session* attribute), 67
parent_rating_key (*tautulli.models.history.Datum* attribute), 83
parent_rating_key (*tautulli.models.library_media_info.Datum* attribute), 91
parent_rating_key (*tautulli.models.metadata.Metadata* attribute), 99
parent_rating_key (*tautulli.models.search_results.EpisodeItem* attribute), 121
parent_rating_key (*tautulli.models.search_results.MovieItem* attribute), 125
parent_rating_key (*tautulli.models.search_results.SeasonItem* attribute), 129
parent_rating_key (*tautulli.models.search_results.ShowItem* attribute), 132
parent_thumb (*tautulli.models.activity.Session* attribute), 67
parent_thumb (*tautulli.models.metadata.Metadata* attribute), 99
parent_thumb (*tautulli.models.search_results.EpisodeItem* attribute), 121
parent_thumb (*tautulli.models.search_results.MovieItem* attribute), 125
parent_thumb (*tautulli.models.search_results.SeasonItem* attribute), 129
parent_thumb (*tautulli.models.search_results.ShowItem* attribute), 132
parent_title (*tautulli.models.activity.Session* attribute), 67
parent_title (*tautulli.models.history.Datum* attribute), 83
parent_title (*tautulli.models.libraries_table.Datum* attribute), 89
parent_title (*tautulli.models.metadata.Metadata* attribute), 99
parent_title (*tautulli.models.search_results.EpisodeItem* attribute), 121
parent_title (*tautulli.models.search_results.MovieItem* attribute), 125
parent_title (*tautulli.models.search_results.SeasonItem* attribute), 129
parent_title (*tautulli.models.search_results.ShowItem* attribute), 132
parent_title (*tautulli.models.user_ips.Datum* attribute), 190
parent_title (*tautulli.models.users_table.Datum* attribute), 188
parent_year (*tautulli.models.metadata.Metadata* attribute), 99
Part (*class in tautulli.models.metadata*), 99
Part (*class in tautulli.models.search_results*), 125
parts (*tautulli.models.metadata.MediaInfoItem* attribute), 96
parts (*tautulli.models.search_results.MediaInfoItem* attribute), 122
paused_counter (*tautulli.models.history.Datum* attribute), 83
percent_complete (*tautulli.models.history.Datum* attribute), 83
photo_quality (*tautulli.models.synced_items.SyncedItems* attribute), 182
ping() (*tautulli.tools.api_helper.APIShortcuts* method), 195
ping_plex() (*tautulli.tools.api_helper.APIShortcuts* method), 195
platform (*tautulli.models.activity.Session* attribute), 67
platform (*tautulli.models.history.Datum* attribute), 83
platform (*tautulli.models.home_stats.Row* attribute), 85
platform (*tautulli.models.pms_update.PMSUpdate* attribute), 85

- tribute), 115
- platform (*tautulli.models.synced_items.SyncedItems* attribute), 182
- platform (*tautulli.models.user_ips.Datum* attribute), 190
- platform (*tautulli.models.user_player_stats.UserPlayerStats* attribute), 192
- platform (*tautulli.models.users_table.Datum* attribute), 188
- platform_name (*tautulli.models.activity.Session* attribute), 67
- platform_name (*tautulli.models.home_stats.Row* attribute), 86
- platform_name (*tautulli.models.user_player_stats.UserPlayerStats* attribute), 192
- platform_version (*tautulli.models.activity.Session* attribute), 67
- play_count (*tautulli.models.library_media_info.Datum* attribute), 91
- play_count (*tautulli.models.user_ips.Datum* attribute), 190
- play_duration (*tautulli.models.history.Datum* attribute), 83
- player (*tautulli.models.activity.Session* attribute), 67
- player (*tautulli.models.history.Datum* attribute), 83
- player (*tautulli.models.home_stats.Row* attribute), 86
- player (*tautulli.models.user_ips.Datum* attribute), 190
- player (*tautulli.models.users_table.Datum* attribute), 188
- player_name (*tautulli.models.user_player_stats.UserPlayerStats* attribute), 192
- PlaylistsTable (class in *tautulli.models.playlists_table*), 114
- playlistType (*tautulli.models.playlists_table.Datum* attribute), 114
- plays (*tautulli.models.libraries_table.Datum* attribute), 89
- plays (*tautulli.models.users_table.Datum* attribute), 188
- Plex (class in *tautulli.models.settings*), 162
- Plex (*tautulli.models.settings.Settings* attribute), 171
- plex_client_host (*tautulli.models.settings.Plex* attribute), 163
- plex_enabled (*tautulli.models.settings.Plex* attribute), 163
- plex_on_buffer (*tautulli.models.settings.Plex* attribute), 163
- plex_on_concurrent (*tautulli.models.settings.Plex* attribute), 163
- plex_on_created (*tautulli.models.settings.Plex* attribute), 163
- plex_on_extdown (*tautulli.models.settings.Plex* attribute), 163
- plex_on_extup (*tautulli.models.settings.Plex* attribute), 163
- plex_on_intdown (*tautulli.models.settings.Plex* attribute), 163
- plex_on_intup (*tautulli.models.settings.Plex* attribute), 163
- plex_on_newdevice (*tautulli.models.settings.Plex* attribute), 163
- plex_on_pause (*tautulli.models.settings.Plex* attribute), 163
- plex_on_play (*tautulli.models.settings.Plex* attribute), 163
- plex_on_pmsupdate (*tautulli.models.settings.Plex* attribute), 163
- plex_on_resume (*tautulli.models.settings.Plex* attribute), 163
- plex_on_stop (*tautulli.models.settings.Plex* attribute), 163
- plex_on_watched (*tautulli.models.settings.Plex* attribute), 163
- plex_password (*tautulli.models.settings.Plex* attribute), 163
- plex_username (*tautulli.models.settings.Plex* attribute), 163
- PlexLog (class in *tautulli.models.plex_log*), 115
- plexpy_auto_update (*tautulli.models.settings.General* attribute), 149
- PlexWatch (class in *tautulli.models.settings*), 163
- PlexWatch (*tautulli.models.settings.Settings* attribute), 171
- plexwatch_database (*tautulli.models.settings.PlexWatch* attribute), 164
- PMS (class in *tautulli.models.settings*), 161
- PMS (*tautulli.models.settings.Settings* attribute), 171
- pms_identifier (*tautulli.models.registered_device.RegisteredDevice* attribute), 117
- pms_identifier (*tautulli.models.server_info.ServerInfo* attribute), 135
- pms_identifier (*tautulli.models.settings.PMS* attribute), 161
- pms_image_proxy (*tautulli.models.docs.Docs* attribute), 79
- pms_image_proxy() (*tautulli.api.json_api.RawAPI* method), 26
- pms_image_proxy() (*tautulli.api.object_api.ObjectAPI* method), 55
- pms_ip (*tautulli.models.registered_device.RegisteredDevice* attribute), 117
- pms_ip (*tautulli.models.server_info.ServerInfo* attribute), 135
- pms_ip (*tautulli.models.settings.PMS* attribute), 161
- pms_is_cloud (*tautulli.models.registered_device.RegisteredDevice* attribute), 117

pms_is_cloud (tautulli.models.settings.PMS attribute), 161
 pms_is_remote (tautulli.models.registered_device.RegisteredDevice attribute), 117
 pms_is_remote (tautulli.models.server_info.ServerInfo attribute), 135
 pms_is_remote (tautulli.models.settings.PMS attribute), 161
 pms_logs_folder (tautulli.models.settings.PMS attribute), 161
 pms_logs_line_cap (tautulli.models.settings.PMS attribute), 161
 pms_name (tautulli.models.registered_device.RegisteredDevice attribute), 117
 pms_name (tautulli.models.server_info.ServerInfo attribute), 135
 pms_name (tautulli.models.settings.PMS attribute), 162
 pms_platform (tautulli.models.registered_device.RegisteredDevice attribute), 117
 pms_platform (tautulli.models.server_info.ServerInfo attribute), 135
 pms_platform (tautulli.models.settings.PMS attribute), 162
 pms_plexpass (tautulli.models.registered_device.RegisteredDevice attribute), 117
 pms_plexpass (tautulli.models.server_info.ServerInfo attribute), 135
 pms_plexpass (tautulli.models.settings.PMS attribute), 162
 pms_port (tautulli.models.registered_device.RegisteredDevice attribute), 117
 pms_port (tautulli.models.server_info.ServerInfo attribute), 135
 pms_port (tautulli.models.settings.PMS attribute), 162
 pms_ssl (tautulli.models.registered_device.RegisteredDevice attribute), 117
 pms_ssl (tautulli.models.server_info.ServerInfo attribute), 136
 pms_ssl (tautulli.models.settings.PMS attribute), 162
 pms_timeout (tautulli.models.settings.Advanced attribute), 139
 pms_token (tautulli.models.settings.PMS attribute), 162
 pms_update (tautulli.api.json_api.RawAPI property), 31
 pms_update (tautulli.api.object_api.ObjectAPI property), 60
 pms_update_channel (tautulli.models.settings.PMS attribute), 162
 pms_update_check_interval (tautulli.models.settings.Advanced attribute), 139
 pms_update_distro (tautulli.models.settings.PMS attribute), 162
 pms_update_distro_build (tautulli.models.settings.PMS attribute), 162
 pms_url (tautulli.models.registered_device.RegisteredDevice attribute), 117
 pms_url (tautulli.models.server_info.ServerInfo attribute), 136
 pms_url (tautulli.models.settings.PMS attribute), 162
 pms_url_manual (tautulli.models.registered_device.RegisteredDevice attribute), 117
 pms_url_manual (tautulli.models.server_info.ServerInfo attribute), 136
 pms_url_manual (tautulli.models.settings.PMS attribute), 162
 pms_url_override (tautulli.models.settings.PMS attribute), 162
 pms_use_bif (tautulli.models.settings.PMS attribute), 162
 pms_vmid (tautulli.models.settings.PMS attribute), 162
 pms_version (tautulli.models.registered_device.RegisteredDevice attribute), 117
 pms_version (tautulli.models.server_info.ServerInfo attribute), 136
 pms_version (tautulli.models.settings.PMS attribute), 162
 pms_web_url (tautulli.models.settings.PMS attribute), 162
 PMSUpdate (class in tautulli.models.pms_update), 115
 port (tautulli.models.server_list.ServerListEntry attribute), 136
 port (tautulli.models.servers_info.ServersInfoEntry attribute), 137
 postal_code (tautulli.models.geo_ip_lookup.GeoIPLookup attribute), 81
 postal_code (tautulli.models.whois_lookup.Net attribute), 194
 pre_tautulli (tautulli.models.stream_data.StreamData attribute), 180
 product (tautulli.models.activity.Session attribute), 67
 product (tautulli.models.history.Datum attribute), 83
 product_version (tautulli.models.activity.Session attribute), 67
 profile (tautulli.models.activity.Session attribute), 67
 progress_marker (tautulli.models.activity.Session property), 67
 progress_percent (tautulli.models.activity.Session attribute), 67
 progress_percentage (tautulli.models.activity.Session property), 67
 Prowl (class in tautulli.models.settings), 164
 Prowl (tautulli.models.settings.Settings attribute), 171
 prowl_enabled (tautulli.models.settings.Prowl attribute), 164
 prowl_keys (tautulli.models.settings.Prowl attribute), 164

`prowl_on_buffer` (`tautulli.models.settings.Prowl` attribute), 164
`prowl_on_concurrent` (`tautulli.models.settings.Prowl` attribute), 164
`prowl_on_created` (`tautulli.models.settings.Prowl` attribute), 164
`prowl_on_extdown` (`tautulli.models.settings.Prowl` attribute), 164
`prowl_on_extup` (`tautulli.models.settings.Prowl` attribute), 164
`prowl_on_intdown` (`tautulli.models.settings.Prowl` attribute), 164
`prowl_on_intup` (`tautulli.models.settings.Prowl` attribute), 164
`prowl_on_newdevice` (`tautulli.models.settings.Prowl` attribute), 164
`prowl_on_pause` (`tautulli.models.settings.Prowl` attribute), 164
`prowl_on_play` (`tautulli.models.settings.Prowl` attribute), 164
`prowl_on_pmsupdate` (`tautulli.models.settings.Prowl` attribute), 165
`prowl_on_resume` (`tautulli.models.settings.Prowl` attribute), 165
`prowl_on_stop` (`tautulli.models.settings.Prowl` attribute), 165
`prowl_on_watched` (`tautulli.models.settings.Prowl` attribute), 165
`prowl_priority` (`tautulli.models.settings.Prowl` attribute), 165
`Pushalot` (class in `tautulli.models.settings`), 166
`Pushalot` (`tautulli.models.settings.Settings` attribute), 171
`pushalot_apikey` (`tautulli.models.settings.Pushalot` attribute), 166
`pushalot_enabled` (`tautulli.models.settings.Pushalot` attribute), 166
`pushalot_on_buffer` (`tautulli.models.settings.Pushalot` attribute), 166
`pushalot_on_concurrent` (`tautulli.models.settings.Pushalot` attribute), 166
`pushalot_on_created` (`tautulli.models.settings.Pushalot` attribute), 166
`pushalot_on_extdown` (`tautulli.models.settings.Pushalot` attribute), 167
`pushalot_on_extup` (`tautulli.models.settings.Pushalot` attribute), 167
`pushalot_on_intdown` (`tautulli.models.settings.Pushalot` attribute), 167
`pushalot_on_intup` (`tautulli.models.settings.Pushalot` attribute), 167
`pushalot_on_newdevice` (`tautulli.models.settings.Pushalot` attribute), 167
`pushalot_on_pause` (`tautulli.models.settings.Pushalot` attribute), 167
`pushalot_on_play` (`tautulli.models.settings.Pushalot` attribute), 167
`pushalot_on_pmsupdate` (`tautulli.models.settings.Pushalot` attribute), 167
`pushalot_on_resume` (`tautulli.models.settings.Pushalot` attribute), 167
`pushalot_on_stop` (`tautulli.models.settings.Pushalot` attribute), 167
`pushalot_on_watched` (`tautulli.models.settings.Pushalot` attribute), 167
`PushBullet` (class in `tautulli.models.settings`), 165
`PushBullet` (`tautulli.models.settings.Settings` attribute), 171
`pushbullet_apikey` (`tautulli.models.settings.PushBullet` attribute), 165
`pushbullet_channel_tag` (`tautulli.models.settings.PushBullet` attribute), 165
`pushbullet_deviceid` (`tautulli.models.settings.PushBullet` attribute), 165
`pushbullet_enabled` (`tautulli.models.settings.PushBullet` attribute), 165
`pushbullet_on_buffer` (`tautulli.models.settings.PushBullet` attribute), 165
`pushbullet_on_concurrent` (`tautulli.models.settings.PushBullet` attribute), 165
`pushbullet_on_created` (`tautulli.models.settings.PushBullet` attribute), 165
`pushbullet_on_extdown` (`tautulli.models.settings.PushBullet` attribute), 165
`pushbullet_on_extup` (`tautulli.models.settings.PushBullet` attribute), 166
`pushbullet_on_intdown` (`tautulli.models.settings.PushBullet` attribute), 166
`pushbullet_on_intup` (`tautulli.models.settings.PushBullet` attribute), 166

<code>tulli.models.settings.PushBullet</code> 166	<code>attribute),</code>	<code>pushover_on_intdown</code> <code>tulli.models.settings.Pushover</code> 168	<code>(tau-</code> <code>attribute),</code>
<code>pushbullet_on_newdevice</code> <code>tulli.models.settings.PushBullet</code> 166	<code>attribute),</code>	<code>pushover_on_intup</code> (<code>tautulli.models.settings.Pushover</code> <code>attribute</code>), 168	
<code>pushbullet_on_pause</code> <code>tulli.models.settings.PushBullet</code> 166	<code>(tau-</code> <code>attribute),</code>	<code>pushover_on_newdevice</code> <code>tulli.models.settings.Pushover</code> 168	<code>(tau-</code> <code>attribute),</code>
<code>pushbullet_on_play</code> <code>tulli.models.settings.PushBullet</code> 166	<code>(tau-</code> <code>attribute),</code>	<code>pushover_on_pause</code> (<code>tautulli.models.settings.Pushover</code> <code>attribute</code>), 168	
<code>pushbullet_on_pmsupdate</code> <code>tulli.models.settings.PushBullet</code> 166	<code>(tau-</code> <code>attribute),</code>	<code>pushover_on_play</code> (<code>tautulli.models.settings.Pushover</code> <code>attribute</code>), 168	
<code>pushbullet_on_resume</code> <code>tulli.models.settings.PushBullet</code> 166	<code>(tau-</code> <code>attribute),</code>	<code>pushover_on_pmsupdate</code> <code>tulli.models.settings.Pushover</code> 168	<code>(tau-</code> <code>attribute),</code>
<code>pushbullet_on_stop</code> <code>tulli.models.settings.PushBullet</code> 166	<code>(tau-</code> <code>attribute),</code>	<code>pushover_on_resume</code> <code>tulli.models.settings.Pushover</code> 168	<code>(tau-</code> <code>attribute),</code>
<code>pushbullet_on_watched</code> <code>tulli.models.settings.PushBullet</code> 166	<code>(tau-</code> <code>attribute),</code>	<code>pushover_on_stop</code> (<code>tautulli.models.settings.Pushover</code> <code>attribute</code>), 168	
<code>Pushover</code> (<i>class in</i> <code>tautulli.models.settings</code>), 167		<code>pushover_on_watched</code> <code>tulli.models.settings.Pushover</code> 168	<code>(tau-</code> <code>attribute),</code>
<code>Pushover</code> (<code>tautulli.models.settings.Settings</code> <code>attribute</code>), 171		<code>pushover_priority</code> (<code>tautulli.models.settings.Pushover</code> <code>attribute</code>), 168	
<code>pushover_apitoken</code> (<code>tautulli.models.settings.Pushover</code> <code>attribute</code>), 167		<code>pushover_sound</code> (<code>tautulli.models.settings.Pushover</code> <code>at-</code> <code>tribute</code>), 168	
<code>pushover_enabled</code> (<code>tautulli.models.settings.Pushover</code> <code>attribute</code>), 168			
<code>pushover_html_support</code> <code>tulli.models.settings.Pushover</code> 168	<code>(tau-</code> <code>attribute),</code>		
<code>pushover_incl_pmslink</code> <code>tulli.models.settings.Pushover</code> 168	<code>(tau-</code> <code>attribute),</code>		
<code>pushover_incl_url</code> (<code>tautulli.models.settings.Pushover</code> <code>attribute</code>), 168			
<code>pushover_keys</code> (<code>tautulli.models.settings.Pushover</code> <code>at-</code> <code>tribute</code>), 168			
<code>pushover_on_buffer</code> <code>tulli.models.settings.Pushover</code> 168	<code>(tau-</code> <code>attribute),</code>		
<code>pushover_on_concurrent</code> <code>tulli.models.settings.Pushover</code> 168	<code>(tau-</code> <code>attribute),</code>		
<code>pushover_on_created</code> <code>tulli.models.settings.Pushover</code> 168	<code>(tau-</code> <code>attribute),</code>		
<code>pushover_on_extdown</code> <code>tulli.models.settings.Pushover</code> 168	<code>(tau-</code> <code>attribute),</code>		
<code>pushover_on_extup</code> (<code>tautulli.models.settings.Pushover</code> <code>attribute</code>), 168			

Q

<code>quality_profile</code> (<code>tautulli.models.activity.Session</code> <code>at-</code> <code>tribute</code>), 67	
<code>quality_profile</code> <code>tulli.models.stream_data.StreamData</code> <code>at-</code> <code>tribute</code>), 180	
<code>query_days</code> (<code>tautulli.models.library_watch_time_stats.LibraryWatchTimeS</code> <code>tribute</code>), 94	
<code>query_days</code> (<code>tautulli.models.user_watch_time_stats.UserWatchTimeStats</code> <code>tribute</code>), 193	

R

<code>range</code> (<code>tautulli.models.whois_lookup.Net</code> <code>attribute</code>), 194	
<code>rating</code> (<code>tautulli.models.activity.Session</code> <code>attribute</code>), 67	
<code>rating</code> (<code>tautulli.models.metadata.Metadata</code> <code>attribute</code>), 99	
<code>rating</code> (<code>tautulli.models.search_results.EpisodeItem</code> <code>at-</code> <code>tribute</code>), 121	
<code>rating</code> (<code>tautulli.models.search_results.MovieItem</code> <code>attribute</code>), 125	
<code>rating</code> (<code>tautulli.models.search_results.SeasonItem</code> <code>at-</code> <code>tribute</code>), 129	
<code>rating</code> (<code>tautulli.models.search_results.ShowItem</code> <code>at-</code> <code>tribute</code>), 132	
<code>rating_image</code> (<code>tautulli.models.activity.Session</code> <code>at-</code> <code>tribute</code>), 67	

`rating_image` (*tautulli.models.metadata.Metadata* attribute), 99
`rating_image` (*tautulli.models.search_results.EpisodeItem* attribute), 121
`rating_image` (*tautulli.models.search_results.MovieItem* attribute), 125
`rating_image` (*tautulli.models.search_results.SeasonItem* attribute), 129
`rating_image` (*tautulli.models.search_results.ShowItem* attribute), 132
`rating_key` (*tautulli.models.activity.Session* attribute), 67
`rating_key` (*tautulli.models.history.Datum* attribute), 83
`rating_key` (*tautulli.models.home_stats.Row* attribute), 86
`rating_key` (*tautulli.models.libraries_table.Datum* attribute), 90
`rating_key` (*tautulli.models.library_media_info.Datum* attribute), 91
`rating_key` (*tautulli.models.metadata.Metadata* attribute), 99
`rating_key` (*tautulli.models.new_rating_keys.Field0* attribute), 102
`rating_key` (*tautulli.models.notification_log.Datum* attribute), 111
`rating_key` (*tautulli.models.old_rating_keys.Field0* attribute), 113
`rating_key` (*tautulli.models.search_results.EpisodeItem* attribute), 121
`rating_key` (*tautulli.models.search_results.MovieItem* attribute), 125
`rating_key` (*tautulli.models.search_results.SeasonItem* attribute), 129
`rating_key` (*tautulli.models.search_results.ShowItem* attribute), 132
`rating_key` (*tautulli.models.synced_items.SyncedItems* attribute), 182
`rating_key` (*tautulli.models.user_ips.Datum* attribute), 190
`rating_key` (*tautulli.models.users_table.Datum* attribute), 188
`rating_keys` (*tautulli.models.settings.GetFilesSizesHold* attribute), 149
`ratingKey` (*tautulli.models.collections_table.Datum* attribute), 73
`ratingKey` (*tautulli.models.playlists_table.Datum* attribute), 114
`RawAPI` (class in *tautulli.api.json_api*), 3
`recently_added` (*tautulli.models.recently_added.RecentlyAdded* attribute), 116
`RecentlyAdded` (class in *tautulli.models.recently_added*), 116
`recordsFiltered` (*tautulli.models.collections_table.CollectionsTable* attribute), 72
`recordsFiltered` (*tautulli.models.history.History* attribute), 84
`recordsFiltered` (*tautulli.models.libraries_table.LibrariesTable* attribute), 90
`recordsFiltered` (*tautulli.models.library_media_info.LibraryMediaInfo* attribute), 92
`recordsFiltered` (*tautulli.models.newsletter_log.NewsletterLog* attribute), 109
`recordsFiltered` (*tautulli.models.notification_log.NotificationLog* attribute), 111
`recordsFiltered` (*tautulli.models.playlists_table.PlaylistsTable* attribute), 114
`recordsFiltered` (*tautulli.models.user_ips.UserIPs* attribute), 190
`recordsFiltered` (*tautulli.models.user_logins.UserLogins* attribute), 191
`recordsFiltered` (*tautulli.models.users_table.UsersTable* attribute), 188
`recordsTotal` (*tautulli.models.collections_table.CollectionsTable* attribute), 72
`recordsTotal` (*tautulli.models.history.History* attribute), 84
`recordsTotal` (*tautulli.models.libraries_table.LibrariesTable* attribute), 90
`recordsTotal` (*tautulli.models.library_media_info.LibraryMediaInfo* attribute), 92
`recordsTotal` (*tautulli.models.newsletter_log.NewsletterLog* attribute), 109
`recordsTotal` (*tautulli.models.notification_log.NotificationLog* attribute), 111
`recordsTotal` (*tautulli.models.playlists_table.PlaylistsTable* attribute), 114
`recordsTotal` (*tautulli.models.user_ips.UserIPs* attribute), 190
`recordsTotal` (*tautulli.models.user_logins.UserLogins* attribute), 191
`recordsTotal` (*tautulli.models.users_table.UsersTable* attribute), 188
`reference_id` (*tautulli.models.history.Datum* attribute), 83
`refresh_libraries_interval` (*tautulli.models.settings.Monitoring* attribute), 158
`refresh_libraries_list` (*tautulli.models.docs.Docs*

attribute), 79
 refresh_libraries_list() (tautulli.api.json_api.RawAPI method), 26
 refresh_libraries_list() (tautulli.api.object_api.ObjectAPI method), 56
 refresh_libraries_on_startup (tautulli.models.settings.Monitoring attribute), 158
 refresh_users_interval (tautulli.models.settings.Monitoring attribute), 158
 refresh_users_list (tautulli.models.docs.Docs attribute), 79
 refresh_users_list() (tautulli.api.json_api.RawAPI method), 26
 refresh_users_list() (tautulli.api.object_api.ObjectAPI method), 56
 refresh_users_on_startup (tautulli.models.settings.Monitoring attribute), 158
 region (tautulli.models.geo_ip_lookup.GeoIPLookup attribute), 81
 register_device (tautulli.models.docs.Docs attribute), 79
 register_device() (tautulli.api.json_api.RawAPI method), 27
 register_device() (tautulli.api.object_api.ObjectAPI method), 56
 RegisteredDevice (class in tautulli.models.registered_device), 116
 regroup_history() (tautulli.api.json_api.RawAPI method), 27
 regroup_history() (tautulli.api.object_api.ObjectAPI method), 57
 relayed (tautulli.models.activity.Session attribute), 67
 release_date (tautulli.models.pms_update.PMSUpdate attribute), 116
 remote_access_ping_interval (tautulli.models.settings.Advanced attribute), 139
 remote_access_ping_threshold (tautulli.models.settings.Advanced attribute), 139
 requirements (tautulli.models.pms_update.PMSUpdate attribute), 116
 restart (tautulli.models.docs.Docs attribute), 79
 restart() (tautulli.api.json_api.RawAPI method), 27
 restart() (tautulli.api.object_api.ObjectAPI method), 57
 result (tautulli.models.server_status.ServerStatus attribute), 137
 result_id (tautulli.models.user_player_stats.UserPlayerStats attribute), 192
 results_count (tautulli.models.search_results.SearchResults attribute), 127
 results_list (tautulli.models.search_results.SearchResults attribute), 127
 ResultsList (class in tautulli.models.search_results), 126
 root_title (tautulli.models.synced_items.SyncedItems attribute), 182
 Row (class in tautulli.models.home_stats), 84
 row_id (tautulli.models.activity.Session attribute), 67
 row_id (tautulli.models.history.Datum attribute), 83
 row_id (tautulli.models.home_stats.Row attribute), 86
 row_id (tautulli.models.libraries_table.Datum attribute), 90
 row_id (tautulli.models.library.Library attribute), 87
 row_id (tautulli.models.user.User attribute), 185
 row_id (tautulli.models.users.User attribute), 186
 row_id (tautulli.models.users_table.Datum attribute), 188
 rows (tautulli.models.home_stats.HomeStat attribute), 84
S
 save_only (tautulli.models.newsletter_config.Config attribute), 103
 Scripts (class in tautulli.models.settings), 168
 Scripts (tautulli.models.settings.Settings attribute), 171
 scripts_enabled (tautulli.models.settings.Scripts attribute), 169
 scripts_folder (tautulli.models.settings.Scripts attribute), 169
 scripts_on_buffer (tautulli.models.settings.Scripts attribute), 169
 scripts_on_buffer_script (tautulli.models.settings.Scripts attribute), 169
 scripts_on_concurrent (tautulli.models.settings.Scripts attribute), 169
 scripts_on_concurrent_script (tautulli.models.settings.Scripts attribute), 169
 scripts_on_created (tautulli.models.settings.Scripts attribute), 169
 scripts_on_created_script (tautulli.models.settings.Scripts attribute), 169
 scripts_on_extdown (tautulli.models.settings.Scripts attribute), 170
 scripts_on_extdown_script (tautulli.models.settings.Scripts attribute), 170
 scripts_on_extup (tautulli.models.settings.Scripts attribute), 170
 scripts_on_extup_script (tautulli.models.settings.Scripts attribute), 170
 scripts_on_intdown (tautulli.models.settings.Scripts attribute), 170

scripts_on_intdown_script (tautulli.models.settings.Scripts attribute), 170
 scripts_on_intup (tautulli.models.settings.Scripts attribute), 170
 scripts_on_intup_script (tautulli.models.settings.Scripts attribute), 170
 scripts_on_newdevice (tautulli.models.settings.Scripts attribute), 170
 scripts_on_newdevice_script (tautulli.models.settings.Scripts attribute), 170
 scripts_on_pause (tautulli.models.settings.Scripts attribute), 170
 scripts_on_pause_script (tautulli.models.settings.Scripts attribute), 170
 scripts_on_play (tautulli.models.settings.Scripts attribute), 170
 scripts_on_play_script (tautulli.models.settings.Scripts attribute), 170
 scripts_on_pmsupdate (tautulli.models.settings.Scripts attribute), 170
 scripts_on_pmsupdate_script (tautulli.models.settings.Scripts attribute), 170
 scripts_on_resume (tautulli.models.settings.Scripts attribute), 170
 scripts_on_resume_script (tautulli.models.settings.Scripts attribute), 170
 scripts_on_stop (tautulli.models.settings.Scripts attribute), 170
 scripts_on_stop_script (tautulli.models.settings.Scripts attribute), 170
 scripts_on_watched (tautulli.models.settings.Scripts attribute), 170
 scripts_on_watched_script (tautulli.models.settings.Scripts attribute), 170
 scripts_timeout (tautulli.models.settings.Scripts attribute), 170
 search (tautulli.models.docs.Docs attribute), 79
 search() (tautulli.api.json_api.RawAPI method), 27
 search() (tautulli.api.object_api.ObjectAPI method), 57
 SearchResults (class in tautulli.models.search_results), 126
 season (tautulli.models.search_results.ResultsList attribute), 126
 SeasonItem (class in tautulli.models.search_results), 127
 section_id (tautulli.models.activity.Session attribute), 67
 section_id (tautulli.models.home_stats.Row attribute), 86
 section_id (tautulli.models.libraries.LibrariesEntry attribute), 88
 section_id (tautulli.models.libraries_table.Datum attribute), 90
 section_id (tautulli.models.library.Library attribute), 87
 section_id (tautulli.models.library_media_info.Datum attribute), 91
 section_id (tautulli.models.library_names.LibraryName attribute), 93
 section_id (tautulli.models.metadata.Metadata attribute), 99
 section_id (tautulli.models.search_results.EpisodeItem attribute), 121
 section_id (tautulli.models.search_results.MovieItem attribute), 125
 section_id (tautulli.models.search_results.SeasonItem attribute), 129
 section_id (tautulli.models.search_results.ShowItem attribute), 132
 section_ids (tautulli.models.settings.GetFilesSizesHold attribute), 149
 section_name (tautulli.models.libraries.LibrariesEntry attribute), 88
 section_name (tautulli.models.libraries_table.Datum attribute), 90
 section_name (tautulli.models.library.Library attribute), 87
 section_name (tautulli.models.library_names.LibraryName attribute), 93
 section_type (tautulli.models.libraries.LibrariesEntry attribute), 88
 section_type (tautulli.models.libraries_table.Datum attribute), 90
 section_type (tautulli.models.library.Library attribute), 87
 section_type (tautulli.models.library_media_info.Datum attribute), 91
 section_type (tautulli.models.library_names.LibraryName attribute), 93
 secure (tautulli.models.activity.Session attribute), 67
 select_options (tautulli.models.newsletter_config.ConfigOption attribute), 104
 select_options (tautulli.models.newsletter_config.EmailConfigOption attribute), 105
 selected (tautulli.models.activity.Session attribute), 67
 selected (tautulli.models.metadata.Part attribute), 100
 selected (tautulli.models.metadata.Stream attribute), 101
 selected (tautulli.models.search_results.Part attribute), 126
 selected (tautulli.models.search_results.Stream attribute), 133
 SelectOption (class in tautulli.models.newsletter_config), 107
 SelectOptions (class in tautulli.models.newsletter_config), 107

[server_friendly_name](#) ([tautulli.api.json_api.RawAPI](#) property), 31
[server_friendly_name](#) ([tautulli.api.object_api.ObjectAPI](#) property), 60
[server_id](#) ([tautulli.models.libraries_table.Datum](#) attribute), 90
[server_id](#) ([tautulli.models.library.Library](#) attribute), 87
[server_id](#) ([tautulli.models.registered_device.RegisteredDevice](#) attribute), 117
[server_identity](#) ([tautulli.api.json_api.RawAPI](#) property), 31
[server_identity](#) ([tautulli.api.object_api.ObjectAPI](#) property), 61
[server_info](#) ([tautulli.api.json_api.RawAPI](#) property), 31
[server_info](#) ([tautulli.api.object_api.ObjectAPI](#) property), 61
[server_list](#) ([tautulli.api.json_api.RawAPI](#) property), 32
[server_list](#) ([tautulli.api.object_api.ObjectAPI](#) property), 61
[server_status](#) ([tautulli.api.json_api.RawAPI](#) property), 32
[server_status](#) ([tautulli.api.object_api.ObjectAPI](#) property), 61
[server_status](#) ([tautulli.models.docs.Docs](#) attribute), 79
[ServerID](#) (class in [tautulli.models.server_id](#)), 134
[ServerIdentity](#) (class in [tautulli.models.server_identity](#)), 135
[ServerInfo](#) (class in [tautulli.models.server_info](#)), 135
[ServerListEntry](#) (class in [tautulli.models.server_list](#)), 136
[servers_info](#) ([tautulli.api.json_api.RawAPI](#) property), 32
[servers_info](#) ([tautulli.api.object_api.ObjectAPI](#) property), 61
[ServersInfoEntry](#) (class in [tautulli.models.servers_info](#)), 137
[ServerStatus](#) (class in [tautulli.models.server_status](#)), 137
[Session](#) (class in [tautulli.models.activity](#)), 63
[session_db_write_attempts](#) ([tautulli.models.settings.Advanced](#) attribute), 139
[session_db_write_attempts](#) ([tautulli.models.settings.Monitoring](#) attribute), 158
[session_id](#) ([tautulli.models.activity.Session](#) attribute), 68
[session_key](#) ([tautulli.models.activity.Session](#) attribute), 68
[session_key](#) ([tautulli.models.history.Datum](#) attribute), 83
[session_key](#) ([tautulli.models.notification_log.Datum](#) attribute), 111
[sessions](#) ([tautulli.models.activity.Activity](#) attribute), 62
[set_mobile_device_config](#) ([tautulli.models.docs.Docs](#) attribute), 79
[set_mobile_device_config\(\)](#) ([tautulli.api.json_api.RawAPI](#) method), 28
[set_mobile_device_config\(\)](#) ([tautulli.api.object_api.ObjectAPI](#) method), 57
[set_newsletter_config](#) ([tautulli.models.docs.Docs](#) attribute), 79
[set_newsletter_config\(\)](#) ([tautulli.api.json_api.RawAPI](#) method), 28
[set_newsletter_config\(\)](#) ([tautulli.api.object_api.ObjectAPI](#) method), 57
[set_notifier_config](#) ([tautulli.models.docs.Docs](#) attribute), 79
[set_notifier_config\(\)](#) ([tautulli.api.json_api.RawAPI](#) method), 28
[set_notifier_config\(\)](#) ([tautulli.api.object_api.ObjectAPI](#) method), 58
[Settings](#) (class in [tautulli.models.settings](#)), 170
[shared_libraries](#) ([tautulli.models.activity.Session](#) attribute), 68
[shared_libraries](#) ([tautulli.models.user.User](#) attribute), 185
[shared_libraries](#) ([tautulli.models.users.User](#) attribute), 186
[shortcuts](#) ([tautulli.api.json_api.RawAPI](#) property), 32
[shortcuts](#) ([tautulli.api.object_api.ObjectAPI](#) property), 61
[show](#) ([tautulli.models.search_results.ResultsList](#) attribute), 126
[show_advanced_settings](#) ([tautulli.models.settings.General](#) attribute), 149
[ShowItem](#) (class in [tautulli.models.search_results](#)), 129
[Slack](#) (class in [tautulli.models.settings](#)), 172
[Slack](#) ([tautulli.models.settings.Settings](#) attribute), 171
[slack_channel](#) ([tautulli.models.settings.Slack](#) attribute), 173
[slack_enabled](#) ([tautulli.models.settings.Slack](#) attribute), 173
[slack_hook](#) ([tautulli.models.settings.Slack](#) attribute), 173
[slack_icon_emoji](#) ([tautulli.models.settings.Slack](#) attribute), 173
[slack_incl_pmslink](#) ([tautulli.models.settings.Slack](#) attribute), 173
[slack_incl_poster](#) ([tautulli.models.settings.Slack](#) attribute), 173

slack_incl_subject (*tautulli.models.settings.Slack attribute*), 173
 slack_on_buffer (*tautulli.models.settings.Slack attribute*), 173
 slack_on_concurrent (*tautulli.models.settings.Slack attribute*), 173
 slack_on_created (*tautulli.models.settings.Slack attribute*), 173
 slack_on_extdown (*tautulli.models.settings.Slack attribute*), 173
 slack_on_extup (*tautulli.models.settings.Slack attribute*), 173
 slack_on_intdown (*tautulli.models.settings.Slack attribute*), 173
 slack_on_intup (*tautulli.models.settings.Slack attribute*), 173
 slack_on_newdevice (*tautulli.models.settings.Slack attribute*), 173
 slack_on_pause (*tautulli.models.settings.Slack attribute*), 173
 slack_on_play (*tautulli.models.settings.Slack attribute*), 174
 slack_on_pmsupdate (*tautulli.models.settings.Slack attribute*), 174
 slack_on_resume (*tautulli.models.settings.Slack attribute*), 174
 slack_on_stop (*tautulli.models.settings.Slack attribute*), 174
 slack_on_watched (*tautulli.models.settings.Slack attribute*), 174
 slack_username (*tautulli.models.settings.Slack attribute*), 174
 smart (*tautulli.models.playlists_table.Datum attribute*), 114
 smtp_password (*tautulli.models.newsletter_config.EmailConfig attribute*), 104
 smtp_port (*tautulli.models.newsletter_config.EmailConfig attribute*), 104
 smtp_server (*tautulli.models.newsletter_config.EmailConfig attribute*), 104
 smtp_user (*tautulli.models.newsletter_config.EmailConfig attribute*), 104
 sort_title (*tautulli.models.activity.Session attribute*), 68
 sort_title (*tautulli.models.library_media_info.Datum attribute*), 91
 sort_title (*tautulli.models.metadata.Metadata attribute*), 99
 sort_title (*tautulli.models.search_results.EpisodeItem attribute*), 121
 sort_title (*tautulli.models.search_results.MovieItem attribute*), 125
 sort_title (*tautulli.models.search_results.SeasonItem attribute*), 129
 sort_title (*tautulli.models.search_results.ShowItem attribute*), 132
 sql (*tautulli.models.docs.Docs attribute*), 79
 sql() (*tautulli.api.json_api.RawAPI method*), 28
 sql() (*tautulli.api.object_api.ObjectAPI method*), 58
 start_date (*tautulli.models.newsletter_log.Datum attribute*), 109
 start_time_offset (*tautulli.models.metadata.Marker attribute*), 95
 started (*tautulli.models.history.Datum attribute*), 83
 started (*tautulli.models.home_stats.Row attribute*), 86
 stat_id (*tautulli.models.home_stats.HomeStat attribute*), 84
 stat_title (*tautulli.models.home_stats.HomeStat attribute*), 84
 stat_type (*tautulli.models.home_stats.HomeStat attribute*), 84
 state (*tautulli.models.activity.Session attribute*), 68
 state (*tautulli.models.history.Datum attribute*), 83
 state (*tautulli.models.synced_items.SyncedItems attribute*), 182
 state (*tautulli.models.whois_lookup.Net attribute*), 194
 status (*tautulli.models.docs.Docs attribute*), 79
 status() (*tautulli.api.json_api.RawAPI method*), 28
 status() (*tautulli.api.object_api.ObjectAPI method*), 58
 status_icon (*tautulli.models.activity.Session property*), 68
 stopped (*tautulli.models.history.Datum attribute*), 83
 stopped (*tautulli.models.home_stats.Row attribute*), 86
 Stream (*class in tautulli.models.metadata*), 100
 Stream (*class in tautulli.models.search_results*), 132
 stream_aspect_ratio (*tautulli.models.activity.Session attribute*), 68
 stream_audio_bitrate (*tautulli.models.activity.Session attribute*), 68
 stream_audio_bitrate (*tautulli.models.stream_data.StreamData attribute*), 180
 stream_audio_bitrate_mode (*tautulli.models.activity.Session attribute*), 68
 stream_audio_channel_layout (*tautulli.models.activity.Session attribute*), 68
 stream_audio_channel_layout_ (*tautulli.models.activity.Session attribute*), 68
 stream_audio_channels (*tautulli.models.activity.Session attribute*), 68
 stream_audio_channels (*tautulli.models.stream_data.StreamData attribute*), 180
 stream_audio_codec (*tautulli.models.activity.Session attribute*), 68
 stream_audio_codec (*tautulli.models.stream_data.StreamData attribute*), 180

<code>stream_audio_decision</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_subtitle_language</code>	(<i>tautulli.models.activity.Session</i> attribute), 68
<code>stream_audio_decision</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180	<code>stream_subtitle_language_code</code>	(<i>tautulli.models.activity.Session</i> attribute), 68
<code>stream_audio_language</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_subtitle_location</code>	(<i>tautulli.models.activity.Session</i> attribute), 68
<code>stream_audio_language_code</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_subtitle_transient</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_audio_sample_rate</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_video_bit_depth</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_bitrate</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_video_bitrate</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_bitrate</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180	<code>stream_video_bitrate</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180
<code>stream_container</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_video_chroma_subsampling</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_container</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180	<code>stream_video_codec</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_container_decision</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_video_codec</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180
<code>stream_container_decision</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180	<code>stream_video_codec_level</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_count</code>	(<i>tautulli.models.activity.Activity</i> attribute), 62	<code>stream_video_color primaries</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_count</code>	(<i>tautulli.models.activity.ActivitySummary</i> attribute), 63	<code>stream_video_color_range</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_count_direct_play</code>	(<i>tautulli.models.activity.Activity</i> attribute), 62	<code>stream_video_color_space</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_count_direct_stream</code>	(<i>tautulli.models.activity.Activity</i> attribute), 62	<code>stream_video_color_trc</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_count_transcode</code>	(<i>tautulli.models.activity.Activity</i> attribute), 62	<code>stream_video_decision</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_duration</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_video_decision</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180
<code>stream_subtitle_codec</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_video_dynamic_range</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_subtitle_codec</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180	<code>stream_video_dynamic_range</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180
<code>stream_subtitle_container</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_video_framerate</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_subtitle_decision</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_video_framerate</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180
<code>stream_subtitle_decision</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180	<code>stream_video_full_resolution</code>	(<i>tautulli.models.activity.Session</i> attribute), 69
<code>stream_subtitle_forced</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_video_full_resolution</code>	(<i>tautulli.models.stream_data.StreamData</i> attribute), 180
<code>stream_subtitle_format</code>	(<i>tautulli.models.activity.Session</i> attribute), 68	<code>stream_video_height</code>	(<i>tautulli.models.activity.Session</i> attribute), 69

stream_video_height (tautulli.models.stream_data.StreamData attribute), 180
 stream_video_language (tautulli.models.activity.Session attribute), 69
 stream_video_language_code (tautulli.models.activity.Session attribute), 69
 stream_video_ref_frames (tautulli.models.activity.Session attribute), 69
 stream_video_resolution (tautulli.models.activity.Session attribute), 69
 stream_video_scan_type (tautulli.models.activity.Session attribute), 69
 stream_video_width (tautulli.models.activity.Session attribute), 69
 stream_video_width (tautulli.models.stream_data.StreamData attribute), 180
 StreamData (class in tautulli.models.stream_data), 178
 streams (tautulli.models.metadata.Part attribute), 100
 streams (tautulli.models.search_results.Part attribute), 126
 studio (tautulli.models.activity.Session attribute), 69
 studio (tautulli.models.metadata.Metadata attribute), 99
 studio (tautulli.models.search_results.EpisodeItem attribute), 121
 studio (tautulli.models.search_results.MovieItem attribute), 125
 studio (tautulli.models.search_results.SeasonItem attribute), 129
 studio (tautulli.models.search_results.ShowItem attribute), 132
 subject (tautulli.models.newsletter_config.NewsletterConfig attribute), 107
 subject_text (tautulli.models.newsletter_log.Datum attribute), 109
 subject_text (tautulli.models.notification_log.Datum attribute), 111
 subtitle_codec (tautulli.models.activity.Session attribute), 69
 subtitle_codec (tautulli.models.search_results.Stream attribute), 133
 subtitle_codec (tautulli.models.stream_data.StreamData attribute), 180
 subtitle_container (tautulli.models.activity.Session attribute), 69
 subtitle_container (tautulli.models.search_results.Stream attribute), 133
 subtitle_decision (tautulli.models.activity.Session attribute), 69
 subtitle_forced (tautulli.models.activity.Session attribute), 69
 subtitle_forced (tautulli.models.search_results.Stream attribute), 133
 subtitle_format (tautulli.models.activity.Session attribute), 69
 subtitle_format (tautulli.models.search_results.Stream attribute), 133
 subtitle_language (tautulli.models.activity.Session attribute), 69
 subtitle_language (tautulli.models.search_results.Stream attribute), 134
 subtitle_language_code (tautulli.models.activity.Session attribute), 69
 subtitle_language_code (tautulli.models.search_results.Stream attribute), 134
 subtitle_location (tautulli.models.activity.Session attribute), 69
 subtitle_location (tautulli.models.search_results.Stream attribute), 134
 subtitles (tautulli.models.activity.Session attribute), 69
 subtitles (tautulli.models.stream_data.StreamData attribute), 180
 subtype (tautulli.models.collections_table.Datum attribute), 73
 success (tautulli.models.newsletter_log.Datum attribute), 109
 success (tautulli.models.notification_log.Datum attribute), 111
 summary (tautulli.models.activity.Activity property), 62
 summary (tautulli.models.activity.Session attribute), 70
 summary (tautulli.models.collections_table.Datum attribute), 73
 summary (tautulli.models.metadata.Metadata attribute), 99
 summary (tautulli.models.playlists_table.Datum attribute), 114
 summary (tautulli.models.search_results.EpisodeItem attribute), 121
 summary (tautulli.models.search_results.MovieItem attribute), 125
 summary (tautulli.models.search_results.SeasonItem attribute), 129
 summary (tautulli.models.search_results.ShowItem attribute), 132
 sync_id (tautulli.models.synced_items.SyncedItems attribute), 182
 sync_title (tautulli.models.synced_items.SyncedItems attribute), 182
 synced_version (tautulli.models.activity.Session attribute), 69

attribute), 70
synced_version (*tautulli.models.stream_data.StreamData attribute*), 180
synced_version_profile (*tautulli.models.activity.Session attribute*), 70
synced_version_profile (*tautulli.models.stream_data.StreamData attribute*), 180
SyncedItems (*class in tautulli.models.synced_items*), 181
synchronous_mode (*tautulli.models.settings.Advanced attribute*), 139
sys_tray_icon (*tautulli.models.settings.General attribute*), 149
system_analytics (*tautulli.models.settings.Advanced attribute*), 139

T

tagline (*tautulli.models.activity.Session attribute*), 70
tagline (*tautulli.models.metadata.Metadata attribute*), 99
tagline (*tautulli.models.search_results.EpisodeItem attribute*), 121
tagline (*tautulli.models.search_results.MovieItem attribute*), 125
tagline (*tautulli.models.search_results.SeasonItem attribute*), 129
tagline (*tautulli.models.search_results.ShowItem attribute*), 132
tautulli.models.activity
 module, 62
tautulli.models.collections_table
 module, 72
tautulli.models.date_formats
 module, 74
tautulli.models.docs
 module, 74
tautulli.models.export_fields
 module, 79
tautulli.models.geo_ip_lookup
 module, 80
tautulli.models.history
 module, 81
tautulli.models.home_stats
 module, 84
tautulli.models.libraries
 module, 87
tautulli.models.libraries_table
 module, 88
tautulli.models.library
 module, 86
tautulli.models.library_media_info
 module, 90
tautulli.models.library_names
 module, 92
tautulli.models.library_user_stats
 module, 93
tautulli.models.library_watch_time_stats
 module, 93
tautulli.models.logs
 module, 94
tautulli.models.metadata
 module, 94
tautulli.models.new_rating_keys
 module, 102
tautulli.models.newsletter
 module, 110
tautulli.models.newsletter_config
 module, 103
tautulli.models.newsletter_log
 module, 108
tautulli.models.notification_log
 module, 110
tautulli.models.notifier_parameters
 module, 112
tautulli.models.notifiers
 module, 112
tautulli.models.old_rating_keys
 module, 113
tautulli.models.playlists_table
 module, 113
tautulli.models.plex_log
 module, 115
tautulli.models.pms_update
 module, 115
tautulli.models.recently_added
 module, 116
tautulli.models.registered_device
 module, 116
tautulli.models.search_results
 module, 118
tautulli.models.server_id
 module, 134
tautulli.models.server_identity
 module, 135
tautulli.models.server_info
 module, 135
tautulli.models.server_list
 module, 136
tautulli.models.server_status
 module, 137
tautulli.models.servers_info
 module, 137
tautulli.models.settings
 module, 138
tautulli.models.stream_data
 module, 178

tautulli.models.synced_items
 module, 181
 tautulli.models.tautulli_info
 module, 183
 tautulli.models.update_check
 module, 183
 tautulli.models.user
 module, 184
 tautulli.models.user_ips
 module, 189
 tautulli.models.user_logins
 module, 190
 tautulli.models.user_player_stats
 module, 192
 tautulli.models.user_watch_time_stats
 module, 193
 tautulli.models.usernames
 module, 192
 tautulli.models.users
 module, 185
 tautulli.models.users_table
 module, 186
 tautulli.models.whois_lookup
 module, 193
 tautulli_branch (tautulli.models.registered_device.RegisteredDevice attribute), 117
 tautulli_branch (tautulli.models.tautulli_info.TautulliInfo attribute), 183
 tautulli_commit (tautulli.models.registered_device.RegisteredDevice attribute), 118
 tautulli_commit (tautulli.models.tautulli_info.TautulliInfo attribute), 183
 tautulli_info (tautulli.api.json_api.RawAPI property), 32
 tautulli_info (tautulli.api.object_api.ObjectAPI property), 61
 tautulli_install_type (tautulli.models.registered_device.RegisteredDevice attribute), 118
 tautulli_install_type (tautulli.models.tautulli_info.TautulliInfo attribute), 183
 tautulli_platform (tautulli.models.registered_device.RegisteredDevice attribute), 118
 tautulli_platform (tautulli.models.tautulli_info.TautulliInfo attribute), 183
 tautulli_platform_device_name (tautulli.models.registered_device.RegisteredDevice attribute), 118
 tautulli_platform_device_name (tautulli.models.tautulli_info.TautulliInfo attribute), 183
 tautulli_platform_linux_distro (tautulli.models.registered_device.RegisteredDevice attribute), 118
 tautulli_platform_linux_distro (tautulli.models.tautulli_info.TautulliInfo attribute), 183
 tautulli_platform_release (tautulli.models.registered_device.RegisteredDevice attribute), 118
 tautulli_platform_release (tautulli.models.tautulli_info.TautulliInfo attribute), 183
 tautulli_platform_version (tautulli.models.registered_device.RegisteredDevice attribute), 118
 tautulli_platform_version (tautulli.models.tautulli_info.TautulliInfo attribute), 183
 tautulli_python_version (tautulli.models.registered_device.RegisteredDevice attribute), 118
 tautulli_python_version (tautulli.models.tautulli_info.TautulliInfo attribute), 183
 tautulli_version (tautulli.models.registered_device.RegisteredDevice attribute), 118
 tautulli_version (tautulli.models.tautulli_info.TautulliInfo attribute), 183
 TautulliInfo (class in tautulli.models.tautulli_info), 183
 Telegram (class in tautulli.models.settings), 174
 Telegram (tautulli.models.settings.Settings attribute), 171
 telegram_bot_token (tautulli.models.settings.Telegram attribute), 174
 telegram_chat_id (tautulli.models.settings.Telegram attribute), 174
 telegram_disable_web_preview (tautulli.models.settings.Telegram attribute), 174
 telegram_enabled (tautulli.models.settings.Telegram attribute), 174
 telegram_html_support (tautulli.models.settings.Telegram attribute), 174
 telegram_incl_poster (tautulli.models.settings.Telegram attribute),

- 175
- telegram_incl_subject (tautulli.models.settings.Telegram attribute), 175
- telegram_on_buffer (tautulli.models.settings.Telegram attribute), 175
- telegram_on_concurrent (tautulli.models.settings.Telegram attribute), 175
- telegram_on_created (tautulli.models.settings.Telegram attribute), 175
- telegram_on_extdown (tautulli.models.settings.Telegram attribute), 175
- telegram_on_extup (tautulli.models.settings.Telegram attribute), 175
- telegram_on_intdown (tautulli.models.settings.Telegram attribute), 175
- telegram_on_intup (tautulli.models.settings.Telegram attribute), 175
- telegram_on_newdevice (tautulli.models.settings.Telegram attribute), 175
- telegram_on_pause (tautulli.models.settings.Telegram attribute), 175
- telegram_on_play (tautulli.models.settings.Telegram attribute), 175
- telegram_on_pmsupdate (tautulli.models.settings.Telegram attribute), 175
- telegram_on_resume (tautulli.models.settings.Telegram attribute), 175
- telegram_on_stop (tautulli.models.settings.Telegram attribute), 175
- telegram_on_watched (tautulli.models.settings.Telegram attribute), 175
- terminate_session (tautulli.models.docs.Docs attribute), 79
- terminate_session() (tautulli.api.json_api.RawAPI method), 29
- terminate_session() (tautulli.api.object_api.ObjectAPI method), 58
- text (tautulli.models.newsletter_config.MovieLibrary attribute), 105
- text (tautulli.models.newsletter_config.MusicLibrary attribute), 106
- text (tautulli.models.newsletter_config.OtherVideoLibrary attribute), 107
- text (tautulli.models.newsletter_config.SelectOption attribute), 107
- text (tautulli.models.newsletter_config.TVShowLibrary attribute), 108
- themoviedb_apikey (tautulli.models.settings.General attribute), 149
- themoviedb_lookup (tautulli.models.settings.General attribute), 149
- thread (tautulli.models.logs.LogEntry attribute), 94
- threaded (tautulli.models.newsletter_config.Config attribute), 103
- throttled (tautulli.models.activity.Session attribute), 70
- thumb (tautulli.models.activity.Session attribute), 70
- thumb (tautulli.models.collections_table.Datum attribute), 73
- thumb (tautulli.models.history.Datum attribute), 83
- thumb (tautulli.models.home_stats.Row attribute), 86
- thumb (tautulli.models.libraries.LibrariesEntry attribute), 88
- thumb (tautulli.models.libraries_table.Datum attribute), 90
- thumb (tautulli.models.library_media_info.Datum attribute), 91
- thumb (tautulli.models.metadata.Metadata attribute), 99
- thumb (tautulli.models.search_results.EpisodeItem attribute), 121
- thumb (tautulli.models.search_results.MovieItem attribute), 125
- thumb (tautulli.models.search_results.SeasonItem attribute), 129
- thumb (tautulli.models.search_results.ShowItem attribute), 132
- thumb (tautulli.models.user_ips.Datum attribute), 190
- thumb (tautulli.models.users.User attribute), 186
- thumb (tautulli.models.users_table.Datum attribute), 188
- time (tautulli.models.logs.LogEntry attribute), 94
- time_format (tautulli.models.date_formats.DateFormats attribute), 74
- time_format (tautulli.models.settings.General attribute), 149
- time_frame (tautulli.models.newsletter_config.Config attribute), 103
- time_frame_units (tautulli.models.newsletter_config.Config attribute), 103
- timestamp (tautulli.models.newsletter_log.Datum attribute), 109
- timestamp (tautulli.models.notification_log.Datum attribute), 111
- timestamp (tautulli.models.user_logins.Datum attribute), 191
- timezone (tautulli.models.geo_ip_lookup.GeoIPLookup attribute), 81
- title (tautulli.models.activity.Session attribute), 70

<code>title</code> (<code>tautulli.models.collections_table.Datum</code> attribute), 73	<code>track</code> (<code>tautulli.models.search_results.ResultsList</code> attribute), 126
<code>title</code> (<code>tautulli.models.history.Datum</code> attribute), 83	<code>transcode_audio_channels</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>title</code> (<code>tautulli.models.home_stats.Row</code> attribute), 86	<code>transcode_audio_codec</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>title</code> (<code>tautulli.models.library_media_info.Datum</code> attribute), 91	<code>transcode_container</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>title</code> (<code>tautulli.models.metadata.Metadata</code> attribute), 99	<code>transcode_count</code> (<code>tautulli.models.activity.ActivitySummary</code> attribute), 63
<code>title</code> (<code>tautulli.models.playlists_table.Datum</code> attribute), 114	<code>transcode_decision</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>title</code> (<code>tautulli.models.search_results.EpisodeItem</code> attribute), 121	<code>transcode_decision</code> (<code>tautulli.models.history.Datum</code> attribute), 83
<code>title</code> (<code>tautulli.models.search_results.MovieItem</code> attribute), 125	<code>transcode_decision</code> (<code>tautulli.models.user_ips.Datum</code> attribute), 190
<code>title</code> (<code>tautulli.models.search_results.SeasonItem</code> attribute), 129	<code>transcode_decision</code> (<code>tautulli.models.users_table.Datum</code> attribute), 188
<code>title</code> (<code>tautulli.models.search_results.ShowItem</code> attribute), 132	<code>transcode_height</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>title</code> (<code>tautulli.models.stream_data.StreamData</code> attribute), 180	<code>transcode_hw_decode</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>title</code> (<code>tautulli.models.users_table.Datum</code> attribute), 188	<code>transcode_hw_decode_title</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>titleSort</code> (<code>tautulli.models.collections_table.Datum</code> attribute), 74	<code>transcode_hw_decoding</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>tls</code> (<code>tautulli.models.newsletter_config.EmailConfig</code> attribute), 104	<code>transcode_hw_decoding</code> (<code>tautulli.models.stream_data.StreamData</code> attribute), 180
<code>to</code> (<code>tautulli.models.newsletter_config.EmailConfig</code> attribute), 104	<code>transcode_hw_encode</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>total_bandwidth</code> (<code>tautulli.models.activity.Activity</code> attribute), 62	<code>transcode_hw_encode_title</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>total_bandwidth</code> (<code>tautulli.models.activity.ActivitySummary</code> attribute), 63	<code>transcode_hw_encoding</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>total_duration</code> (<code>tautulli.models.history.History</code> attribute), 84	<code>transcode_hw_encoding</code> (<code>tautulli.models.stream_data.StreamData</code> attribute), 180
<code>total_duration</code> (<code>tautulli.models.home_stats.Row</code> attribute), 86	<code>transcode_hw_full_pipeline</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>total_file_size</code> (<code>tautulli.models.library_media_info.LibraryMediaInfo</code> attribute), 92	<code>transcode_hw_full_pipeline</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>total_plays</code> (<code>tautulli.models.home_stats.Row</code> attribute), 86	<code>transcode_key</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>total_plays</code> (<code>tautulli.models.library_user_stats.LibraryUserStats</code> attribute), 93	<code>transcode_max_offset_available</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>total_plays</code> (<code>tautulli.models.library_watch_time_stats.LibraryWatchTimeStats</code> attribute), 94	<code>transcode_min_offset_available</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>total_plays</code> (<code>tautulli.models.user_player_stats.UserPlayerStats</code> attribute), 192	<code>transcode_progress</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>total_plays</code> (<code>tautulli.models.user_watch_time_stats.UserWatchTimeStats</code> attribute), 193	<code>transcode_protocol</code> (<code>tautulli.models.activity.Session</code> attribute), 70
<code>total_size</code> (<code>tautulli.models.synced_items.SyncedItems</code> attribute), 182	
<code>total_time</code> (<code>tautulli.models.library_watch_time_stats.LibraryWatchTimeStats</code> attribute), 94	
<code>total_time</code> (<code>tautulli.models.user_watch_time_stats.UserWatchTimeStats</code> attribute), 193	

- transcode_speed (*tautulli.models.activity.Session attribute*), 70
- transcode_throttled (*tautulli.models.activity.Session attribute*), 70
- transcode_video_codec (*tautulli.models.activity.Session attribute*), 70
- transcode_width (*tautulli.models.activity.Session attribute*), 70
- transcoding_stub (*tautulli.models.activity.Session property*), 70
- tv_logging_enable (*tautulli.models.settings.Monitoring attribute*), 158
- tv_notify_enable (*tautulli.models.settings.Monitoring attribute*), 158
- tv_notify_on_pause (*tautulli.models.settings.Monitoring attribute*), 158
- tv_notify_on_start (*tautulli.models.settings.Monitoring attribute*), 158
- tv_notify_on_stop (*tautulli.models.settings.Monitoring attribute*), 158
- TV_Show_Libraries (*tautulli.models.newsletter_config.SelectOptions attribute*), 107
- tv_watched_percent (*tautulli.models.settings.Monitoring attribute*), 158
- tvmaze_lookup (*tautulli.models.settings.General attribute*), 149
- TVShowLibrary (*class in tautulli.models.newsletter_config*), 108
- Twitter (*class in tautulli.models.settings*), 175
- Twitter (*tautulli.models.settings.Settings attribute*), 171
- twitter_access_token (*tautulli.models.settings.Twitter attribute*), 176
- twitter_access_token_secret (*tautulli.models.settings.Twitter attribute*), 176
- twitter_consumer_key (*tautulli.models.settings.Twitter attribute*), 176
- twitter_consumer_secret (*tautulli.models.settings.Twitter attribute*), 176
- twitter_enabled (*tautulli.models.settings.Twitter attribute*), 176
- twitter_incl_poster (*tautulli.models.settings.Twitter attribute*), 176
- twitter_incl_subject (*tautulli.models.settings.Twitter attribute*), 176
- twitter_on_buffer (*tautulli.models.settings.Twitter attribute*), 176
- twitter_on_concurrent (*tautulli.models.settings.Twitter attribute*), 176
- twitter_on_created (*tautulli.models.settings.Twitter attribute*), 176
- twitter_on_extdown (*tautulli.models.settings.Twitter attribute*), 176
- twitter_on_extup (*tautulli.models.settings.Twitter attribute*), 176
- twitter_on_intdown (*tautulli.models.settings.Twitter attribute*), 176
- twitter_on_intup (*tautulli.models.settings.Twitter attribute*), 176
- twitter_on_newdevice (*tautulli.models.settings.Twitter attribute*), 176
- twitter_on_pause (*tautulli.models.settings.Twitter attribute*), 176
- twitter_on_play (*tautulli.models.settings.Twitter attribute*), 177
- twitter_on_pmsupdate (*tautulli.models.settings.Twitter attribute*), 177
- twitter_on_resume (*tautulli.models.settings.Twitter attribute*), 177
- twitter_on_stop (*tautulli.models.settings.Twitter attribute*), 177
- twitter_on_watched (*tautulli.models.settings.Twitter attribute*), 177
- type (*tautulli.models.activity.Session attribute*), 70
- type (*tautulli.models.collections_table.Datum attribute*), 74
- type (*tautulli.models.metadata.Marker attribute*), 95
- type (*tautulli.models.metadata.Stream attribute*), 101
- type (*tautulli.models.notifier_parameters.NotifierParameter attribute*), 112
- type (*tautulli.models.playlists_table.Datum attribute*), 114
- type (*tautulli.models.search_results.Stream attribute*), 134
- type_icon (*tautulli.models.activity.Session property*), 71
- ## U
- undelete_library (*tautulli.models.docs.Docs attribute*), 79
- undelete_library() (*tautulli.api.json_api.RawAPI method*), 29
- undelete_library() (*tautulli.api.object_api.ObjectAPI method*), 59
- undelete_user (*tautulli.models.docs.Docs attribute*), 79
- undelete_user() (*tautulli.api.json_api.RawAPI method*), 29
- undelete_user() (*tautulli.api.object_api.ObjectAPI method*), 59
- update (*tautulli.models.docs.Docs attribute*), 79

- `update` (*tautulli.models.update_check.UpdateCheck* attribute), 184
- `update()` (*tautulli.api.json_api.RawAPI* method), 29
- `update()` (*tautulli.api.object_api.ObjectAPI* method), 59
- `update_available` (*tautulli.models.pms_update.PMSUpdate* attribute), 116
- `update_check` (*tautulli.api.json_api.RawAPI* property), 32
- `update_check` (*tautulli.api.object_api.ObjectAPI* property), 61
- `update_check` (*tautulli.models.docs.Docs* attribute), 79
- `update_db_interval` (*tautulli.models.settings.General* attribute), 149
- `update_labels` (*tautulli.models.settings.General* attribute), 149
- `update_libraries_db_notify` (*tautulli.models.settings.General* attribute), 149
- `update_metadata_details` (*tautulli.models.docs.Docs* attribute), 79
- `update_metadata_details()` (*tautulli.api.json_api.RawAPI* method), 30
- `update_metadata_details()` (*tautulli.api.object_api.ObjectAPI* method), 59
- `update_notifiers_db` (*tautulli.models.settings.General* attribute), 149
- `update_section_ids` (*tautulli.models.settings.General* attribute), 149
- `update_show_changelog` (*tautulli.models.settings.General* attribute), 149
- `UpdateCheck` (class in *tautulli.models.update_check*), 183
- `updated` (*tautulli.models.whois_lookup.Net* attribute), 194
- `updated_at` (*tautulli.models.activity.Session* attribute), 71
- `updated_at` (*tautulli.models.metadata.Metadata* attribute), 99
- `updated_at` (*tautulli.models.search_results.EpisodeItem* attribute), 121
- `updated_at` (*tautulli.models.search_results.MovieItem* attribute), 125
- `updated_at` (*tautulli.models.search_results.SeasonItem* attribute), 129
- `updated_at` (*tautulli.models.search_results.ShowItem* attribute), 132
- `updatedAt` (*tautulli.models.collections_table.Datum* attribute), 74
- `updatedAt` (*tautulli.models.playlists_table.Datum* attribute), 114
- `uri` (*tautulli.models.server_list.ServerListEntry* attribute), 136
- `User` (class in *tautulli.models.user*), 184
- `User` (class in *tautulli.models.users*), 185
- `user` (*tautulli.models.activity.Session* attribute), 71
- `user` (*tautulli.models.history.Datum* attribute), 83
- `user` (*tautulli.models.home_stats.Row* attribute), 86
- `user` (*tautulli.models.notification_log.Datum* attribute), 111
- `user` (*tautulli.models.synced_items.SyncedItems* attribute), 182
- `user` (*tautulli.models.user_logins.Datum* attribute), 191
- `user_agent` (*tautulli.models.user_logins.Datum* attribute), 191
- `user_group` (*tautulli.models.user_logins.Datum* attribute), 191
- `user_id` (*tautulli.models.activity.Session* attribute), 71
- `user_id` (*tautulli.models.history.Datum* attribute), 83
- `user_id` (*tautulli.models.home_stats.Row* attribute), 86
- `user_id` (*tautulli.models.library_user_stats.LibraryUserStats* attribute), 93
- `user_id` (*tautulli.models.notification_log.Datum* attribute), 111
- `user_id` (*tautulli.models.synced_items.SyncedItems* attribute), 182
- `user_id` (*tautulli.models.user.User* attribute), 185
- `user_id` (*tautulli.models.user_ips.Datum* attribute), 190
- `user_id` (*tautulli.models.user_logins.Datum* attribute), 191
- `user_id` (*tautulli.models.usernames.UserName* attribute), 192
- `user_id` (*tautulli.models.users.User* attribute), 186
- `user_id` (*tautulli.models.users_table.Datum* attribute), 188
- `user_names` (*tautulli.api.json_api.RawAPI* property), 32
- `user_names` (*tautulli.api.object_api.ObjectAPI* property), 61
- `user_rating` (*tautulli.models.activity.Session* attribute), 71
- `user_rating` (*tautulli.models.metadata.Metadata* attribute), 99
- `user_rating` (*tautulli.models.search_results.EpisodeItem* attribute), 121
- `user_rating` (*tautulli.models.search_results.MovieItem* attribute), 125
- `user_rating` (*tautulli.models.search_results.SeasonItem* attribute), 129
- `user_rating` (*tautulli.models.search_results.ShowItem* attribute), 132
- `user_thumb` (*tautulli.models.activity.Session* attribute), 71
- `user_thumb` (*tautulli.models.history.Datum* attribute), 83
- `user_thumb` (*tautulli.models.home_stats.Row* attribute), 86
- `user_thumb` (*tautulli.models.library_user_stats.LibraryUserStats* attribute), 93

- `user_thumb` (*tautulli.models.user.User* attribute), 185
 - `user_thumb` (*tautulli.models.users_table.Datum* attribute), 188
 - `userID` (*tautulli.models.playlists_table.Datum* attribute), 114
 - `UserIPs` (class in *tautulli.models.user_ips*), 190
 - `UserLogins` (class in *tautulli.models.user_logins*), 191
 - `UserName` (class in *tautulli.models.usernames*), 192
 - `username` (*tautulli.models.activity.Session* attribute), 71
 - `username` (*tautulli.models.library_user_stats.LibraryUserStats* attribute), 93
 - `username` (*tautulli.models.synced_items.SyncedItems* attribute), 182
 - `username` (*tautulli.models.user.User* attribute), 185
 - `username` (*tautulli.models.users.User* attribute), 186
 - `username` (*tautulli.models.users_table.Datum* attribute), 188
 - `UserPlayerStats` (class in *tautulli.models.user_player_stats*), 192
 - `users` (*tautulli.api.json_api.RawAPI* property), 33
 - `users` (*tautulli.api.object_api.ObjectAPI* property), 62
 - `users_watched` (*tautulli.models.home_stats.Row* attribute), 86
 - `UsersTable` (class in *tautulli.models.users_table*), 188
 - `UserWatchTimeStats` (class in *tautulli.models.user_watch_time_stats*), 193
 - `uuid` (*tautulli.models.newsletter_log.Datum* attribute), 109
- V**
- `value` (*tautulli.models.newsletter_config.ConfigOption* attribute), 104
 - `value` (*tautulli.models.newsletter_config.EmailConfigOption* attribute), 105
 - `value` (*tautulli.models.newsletter_config.MovieLibrary* attribute), 105
 - `value` (*tautulli.models.newsletter_config.MusicLibrary* attribute), 106
 - `value` (*tautulli.models.newsletter_config.OtherVideoLibrary* attribute), 107
 - `value` (*tautulli.models.newsletter_config.SelectOption* attribute), 107
 - `value` (*tautulli.models.newsletter_config.TVShowLibrary* attribute), 108
 - `value` (*tautulli.models.notifier_parameters.NotifierParameter* attribute), 112
 - `value` (*tautulli.models.server_list.ServerListEntry* attribute), 136
 - `verbose_logs` (*tautulli.models.settings.Advanced* attribute), 139
 - `verify_ssl_cert` (*tautulli.models.settings.Advanced* attribute), 139
 - `version` (*tautulli.models.pms_update.PMSUpdate* attribute), 116
 - `version` (*tautulli.models.server_identity.ServerIdentity* attribute), 135
 - `version` (*tautulli.models.servers_info.ServersInfoEntry* attribute), 137
 - `video_bit_depth` (*tautulli.models.activity.Session* attribute), 71
 - `video_bit_depth` (*tautulli.models.metadata.Stream* attribute), 101
 - `video_bit_depth` (*tautulli.models.search_results.Stream* attribute), 134
 - `video_bitrate` (*tautulli.models.activity.Session* attribute), 71
 - `video_bitrate` (*tautulli.models.metadata.Stream* attribute), 101
 - `video_bitrate` (*tautulli.models.search_results.Stream* attribute), 134
 - `video_bitrate` (*tautulli.models.stream_data.StreamData* attribute), 180
 - `video_bitrate` (*tautulli.models.synced_items.SyncedItems* attribute), 182
 - `video_chroma_subsampling` (*tautulli.models.activity.Session* attribute), 71
 - `video_chroma_subsampling` (*tautulli.models.metadata.Stream* attribute), 101
 - `video_chroma_subsampling` (*tautulli.models.search_results.Stream* attribute), 134
 - `video_codec` (*tautulli.models.activity.Session* attribute), 71
 - `video_codec` (*tautulli.models.library_media_info.Datum* attribute), 92
 - `video_codec` (*tautulli.models.metadata.MediaInfoItem* attribute), 96
 - `video_codec` (*tautulli.models.metadata.Stream* attribute), 101
 - `video_codec` (*tautulli.models.search_results.MediaInfoItem* attribute), 122
 - `video_codec` (*tautulli.models.search_results.Stream* attribute), 134
 - `video_codec` (*tautulli.models.stream_data.StreamData* attribute), 181
 - `video_codec_level` (*tautulli.models.activity.Session* attribute), 71
 - `video_codec_level` (*tautulli.models.metadata.Stream* attribute), 101
 - `video_codec_level` (*tautulli.models.search_results.Stream* attribute), 134
 - `video_color primaries` (*tautulli.models.activity.Session* attribute), 71
 - `video_color primaries` (*tautulli.models.metadata.Stream* attribute),

101	tautulli.models.search_results.Stream attribute), 122	tautulli.models.search_results.MediaInfoItem attribute), 122	
video_color_primaries	(tautulli.models.activity.Session attribute), 71	video_framerate	(tautulli.models.stream_data.StreamData attribute), 181
134		video_full_resolution	(tautulli.models.activity.Session attribute), 71
video_color_range	(tautulli.models.metadata.Stream attribute), 101	video_full_resolution	(tautulli.models.metadata.MediaInfoItem attribute), 96
video_color_range	(tautulli.models.search_results.Stream attribute), 134	video_full_resolution	(tautulli.models.search_results.MediaInfoItem attribute), 122
video_color_space	(tautulli.models.activity.Session attribute), 71	video_full_resolution	(tautulli.models.stream_data.StreamData attribute), 181
video_color_space	(tautulli.models.metadata.Stream attribute), 101	video_height	(tautulli.models.activity.Session attribute), 71
video_color_space	(tautulli.models.search_results.Stream attribute), 134	video_height	(tautulli.models.metadata.Stream attribute), 102
video_color_trc	(tautulli.models.activity.Session attribute), 71	video_height	(tautulli.models.search_results.Stream attribute), 134
video_color_trc	(tautulli.models.metadata.Stream attribute), 101	video_height	(tautulli.models.stream_data.StreamData attribute), 181
video_color_trc	(tautulli.models.search_results.Stream attribute), 134	video_language	(tautulli.models.activity.Session attribute), 71
video_decision	(tautulli.models.activity.Session attribute), 71	video_language	(tautulli.models.metadata.Stream attribute), 102
video_decision	(tautulli.models.stream_data.StreamData attribute), 181	video_language	(tautulli.models.search_results.Stream attribute), 134
video_dynamic_range	(tautulli.models.activity.Session attribute), 71	video_language_code	(tautulli.models.activity.Session attribute), 71
video_dynamic_range	(tautulli.models.metadata.Stream attribute), 102	video_language_code	(tautulli.models.metadata.Stream attribute), 102
video_dynamic_range	(tautulli.models.stream_data.StreamData attribute), 181	video_language_code	(tautulli.models.search_results.Stream attribute), 134
video_frame_rate	(tautulli.models.activity.Session attribute), 71	video_logging_enable	(tautulli.models.settings.Monitoring attribute), 158
video_frame_rate	(tautulli.models.metadata.Stream attribute), 102	video_profile	(tautulli.models.activity.Session attribute), 71
video_frame_rate	(tautulli.models.search_results.Stream attribute), 134	video_profile	(tautulli.models.metadata.MediaInfoItem attribute), 96
video_framerate	(tautulli.models.activity.Session attribute), 71	video_profile	(tautulli.models.metadata.Stream attribute), 102
video_framerate	(tautulli.models.library_media_info.Datum attribute), 92	video_profile	(tautulli.models.search_results.MediaInfoItem attribute), 122
video_framerate	(tautulli.models.metadata.MediaInfoItem attribute), 96	video_profile	(tautulli.models.search_results.Stream attribute), 134
video_framerate	(tautulli.models.activity.Session attribute), 71	video_quality	(tautulli.models.synced_items.SyncedItems attribute), 182
video_framerate	(tautulli.models.activity.Session attribute), 71	video_ref_frames	(tautulli.models.activity.Session attribute), 71

tribute), 71
video_ref_frames (*tautulli.models.metadata.Stream attribute*), 102
video_ref_frames (*tautulli.models.search_results.Stream attribute*), 134
video_resolution (*tautulli.models.activity.Session attribute*), 71
video_resolution (*tautulli.models.library_media_info.Datum attribute*), 92
video_resolution (*tautulli.models.metadata.MediaInfoItem attribute*), 96
video_resolution (*tautulli.models.search_results.MediaInfoItem attribute*), 122
video_scan_type (*tautulli.models.activity.Session attribute*), 71
video_scan_type (*tautulli.models.metadata.Stream attribute*), 102
video_scan_type (*tautulli.models.search_results.Stream attribute*), 134
video_width (*tautulli.models.activity.Session attribute*), 71
video_width (*tautulli.models.metadata.Stream attribute*), 102
video_width (*tautulli.models.search_results.Stream attribute*), 134
video_width (*tautulli.models.stream_data.StreamData attribute*), 181
view_offset (*tautulli.models.activity.Session attribute*), 71

W

wan_bandwidth (*tautulli.models.activity.Activity attribute*), 62
watched_status (*tautulli.models.history.Datum attribute*), 83
websocket_connection_attempts (*tautulli.models.settings.Advanced attribute*), 139
websocket_connection_timeout (*tautulli.models.settings.Advanced attribute*), 139
websocket_monitor_ping_pong (*tautulli.models.settings.Advanced attribute*), 139
week_start_monday (*tautulli.models.settings.General attribute*), 149
WHOISLookup (*class in tautulli.models.whois_lookup*), 194
width (*tautulli.models.activity.Session attribute*), 71

width (*tautulli.models.metadata.MediaInfoItem attribute*), 96
width (*tautulli.models.search_results.MediaInfoItem attribute*), 122
win_sys_tray (*tautulli.models.settings.General attribute*), 149
writers (*tautulli.models.activity.Session attribute*), 72
writers (*tautulli.models.metadata.Metadata attribute*), 99
writers (*tautulli.models.search_results.EpisodeItem attribute*), 121
writers (*tautulli.models.search_results.MovieItem attribute*), 125
writers (*tautulli.models.search_results.SeasonItem attribute*), 129
writers (*tautulli.models.search_results.ShowItem attribute*), 132

X

XBMC (*class in tautulli.models.settings*), 177
XBMC (*tautulli.models.settings.Settings attribute*), 171
xbmc_enabled (*tautulli.models.settings.XBMC attribute*), 177
xbmc_host (*tautulli.models.settings.XBMC attribute*), 177
xbmc_on_buffer (*tautulli.models.settings.XBMC attribute*), 177
xbmc_on_concurrent (*tautulli.models.settings.XBMC attribute*), 177
xbmc_on_created (*tautulli.models.settings.XBMC attribute*), 177
xbmc_on_extdown (*tautulli.models.settings.XBMC attribute*), 177
xbmc_on_extup (*tautulli.models.settings.XBMC attribute*), 177
xbmc_on_intdown (*tautulli.models.settings.XBMC attribute*), 177
xbmc_on_intup (*tautulli.models.settings.XBMC attribute*), 177
xbmc_on_newdevice (*tautulli.models.settings.XBMC attribute*), 178
xbmc_on_pause (*tautulli.models.settings.XBMC attribute*), 178
xbmc_on_play (*tautulli.models.settings.XBMC attribute*), 178
xbmc_on_pmsupdate (*tautulli.models.settings.XBMC attribute*), 178
xbmc_on_resume (*tautulli.models.settings.XBMC attribute*), 178
xbmc_on_stop (*tautulli.models.settings.XBMC attribute*), 178
xbmc_on_watched (*tautulli.models.settings.XBMC attribute*), 178

`xbmc_password` (*tautulli.models.settings.XBMC attribute*), 178
`xbmc_username` (*tautulli.models.settings.XBMC attribute*), 178

Y

`year` (*tautulli.models.activity.Session attribute*), 72
`year` (*tautulli.models.history.Datum attribute*), 83
`year` (*tautulli.models.libraries_table.Datum attribute*), 90
`year` (*tautulli.models.library_media_info.Datum attribute*), 92
`year` (*tautulli.models.metadata.Metadata attribute*), 99
`year` (*tautulli.models.search_results.EpisodeItem attribute*), 121
`year` (*tautulli.models.search_results.MovieItem attribute*), 125
`year` (*tautulli.models.search_results.SeasonItem attribute*), 129
`year` (*tautulli.models.search_results.ShowItem attribute*), 132
`year` (*tautulli.models.user_ips.Datum attribute*), 190
`year` (*tautulli.models.users_table.Datum attribute*), 188