Package: tRakt (via r-universe)

October 10, 2026

Type Package

Title Get Data from 'trakt.tv'

Version 0.19.0.9000

Description A wrapper for the <https://trakt.tv> API to retrieve data about shows and movies, including user ratings, credits and related metadata. Additional functions retrieve user-specific information including collections and history of watched items. A full API reference is available at <https://trakt.docs.apiary.io>.

License MIT + file LICENSE

URL https://jemus42.github.io/tRakt, https://github.com/jemus42/tRakt

BugReports http://github.com/jemus42/tRakt/issues

Depends R (>= 4.1)

Imports cli, curl, dplyr, httr2, lifecycle, lubridate, purrr, rappdirs, rlang, stringr, tibble, utils

Suggests covr, ggplot2, glue, kableExtra, knitr, rmarkdown, roxygen2, scales, testthat (>= 3.0.0), tidyr, vcr (>= 2.0.0), withr, yaml

VignetteBuilder knitr

Config/Needs/website pkgdown

Config/roxygen2/version 8.0.0

Config/testthat/edition 3

Config/testthat/parallel true

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

Config/pak/sysreqs libicu-dev libssl-dev

Repository https://jemus42.r-universe.dev

Date/Publication 2026-09-02 11:58:35 UTC

RemoteUrl https://github.com/jemus42/tRakt

RemoteRef HEAD

RemoteSha 4a3b25368b087fa993c1096210a4197e21946c22

Contents

anticipated_media . . . 3
build_trakt_url . . . 6
collected_media . . . 7
comments_comment . . . 9
comments_trending . . . 11
comments_updates . . . 13
episodes_summary . . . 14
filters . . . 15
futurama . . . 18
gameofthrones . . . 19
lists_popular . . . 20
media_aliases . . . 21
media_comments . . . 22
media_lists . . . 24
media_people . . . 27
media_ratings . . . 29
media_stats . . . 30
media_translations . . . 32
media_watching . . . 33
movies_boxoffice . . . 35
movies_related . . . 36
movies_releases . . . 37
movies_summary . . . 38
pad_episode . . . 39
people_media . . . 40
people_summary . . . 41
played_media . . . 43
popular_media . . . 45
search_query . . . 48
seasons_episodes . . . 51
seasons_season . . . 52
seasons_summary . . . 53
shows_next_episode . . . 55
shows_related . . . 56
shows_summary . . . 57
trakt_api_available . . . 59
trakt_credentials . . . 59
trakt_datasets . . . 60
trakt_get . . . 61
tRakt_sitrep . . . 62
trending_media . . . 62
updated_media . . . 65

user_collection 67
user_comments 68
user_history 69
user_likes 71
user_list 72
user_list_comments 73
user_list_items 74
user_lists 75
user_network 77
user_profile 78
user_ratings 79
user_stats 81
user_watched 82
user_watchlist 83
watched_media 84

Index **88**

`anticipated_media` *Anticipated media*

Description

These functions return the most anticipated movies/shows on trakt.tv.

Usage

```
movies_anticipated(
  limit = 10,
  extended = "min",
  filters = NULL,
  query = NULL,
  years = NULL,
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL
)

shows_anticipated(
  limit = 10,
  extended = "min",
  filters = NULL,
  query = NULL,
  years = NULL,
  genres = NULL,
```

```
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL,
  networks = NULL,
  status = NULL
)
```

Arguments

limit	integer(1) [10L]: Number of items to return. Must be greater than 0 and will be coerced via as.integer().
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).
filters	A trakt_filters object created with filters_movies(), filters_shows(), or filters_episodes() that refines which items are returned. See filters for the full set of supported filters. Supplying filters as individual arguments (genres, years, networks, ...) is soft-deprecated in favour of this argument; if both are given, filters takes precedence.
query	character(1): Search string for titles and descriptions. For search_query() other fields are searched depending on the type of media. See the API docs for a full reference.
years	character | integer: 4-digit year (2010) **or** range, e.g. "2010-2020". Can also be an integer vector of length two which will be coerced appropriately, e.g. c(2010, 2020).
genres	character(n): Genre slug(s). See trakt_genres for a table of genres. Multiple values are allowed and will be concatenated.
languages	character(n): Two-letter language code(s). Also see trakt_languages for available languages (code and name).
countries	character(n): Two-letter country code(s). See trakt_countries.
runtimes	character | integer: Integer range in minutes, e.g. 30-90. Can also be an integer vector of length two which will be coerced appropriately.
ratings	character | integer: Integer range between 0 and 100. Can also be an integer vector of length two which will be coerced appropriately. Note that user-supplied ratings are in the range of 1 to 10, yet the ratings on the site itself are scaled to the range of 1 to 100.

`certifications` `character(n)`: Certification(s) like `pg-13`. Multiple values are allowed. Use `trakt_certifications` for reference. Note that there are different certifications for shows and movies.

`networks` `character(n)`: (Shows only) Network name like `HBO`. See `trakt_networks` for a list of known networks.

`status` `character(n)`: (Shows only) The status of the shows. One of `"returning series"`, `"in production"`, `"planned"`, `"canceled"`, or `"ended"`.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

The Dynamic Lists on trakt.tv

These functions access the automatically updated lists provided by trakt.tv. Each function comes in two flavors: Shows or movies. The following descriptions are adapted directly from the API reference.

- Popular: Popularity is calculated using the rating percentage and the number of ratings.
- Trending: Returns all movies/shows being watched right now. Movies/shows with the most users are returned first.
- Played: Returns the most played (a single user can watch multiple times) movies/shows in the specified time `period`.
- Watched: Returns the most watched (unique users) movies/shows in the specified time `period`.
- Collected: Returns the most collected (unique users) movies/shows in the specified time `period`.
- Anticipated: Returns the most anticipated movies/shows based on the number of lists a movie/show appears on. The functions for **Played**, **Watched**, **Collected** and **Played** each return the same additional variables besides the media information: `watcher_count`, `play_count`, `collected_count`, `collector_count`.

Source

`movies_anticipated()` wraps endpoint /movies/anticipated.

`shows_anticipated()` wraps endpoint /shows/anticipated.

See Also

Other movie data: `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `movies_boxoffice()`, `movies_related()`, `movies_releases()`, `movies_summary()`, `people_media()`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other dynamic lists: `collected_media`, `filters`, `lists_popular()`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other shows data: `popular_media`, `trending_media`, `watched_media`

Examples

```
# Get 15 the most anticipated upcoming shows on Netflix that air this year
current_year <- format(Sys.Date(), "%Y")
shows_anticipated(
  limit = 15,
  filters = filters_shows(networks = "Netflix", years = current_year)
)
```

`build_trakt_url` *Assemble a trakt.tv API URL*

Description

`build_trakt_url` assembles a trakt.tv API URL from different arguments. The result should be fine for use with trakt_get, since that's what this function was created for.

Usage

```
build_trakt_url(...)
```

Arguments

`...`	Unnamed arguments will be concatenated with `/` as separators to form the path of the API URL, e.g. the arguments `movies`, `tron-legacy-2012` and `releases` will be concatenated to `movies/tron-legacy-2012/releases`. Additional **named** arguments will be used as query parameters, usually `extended = "full"` or others.

Value

A URL: `character` of length 1.

See Also

Other utility functions: `pad_episode()`

Examples

```
build_trakt_url("shows", "breaking-bad", extended = "full")
build_trakt_url("shows", "popular", page = 3, limit = 5)

# Path can also be partially assembled already
build_trakt_url("users/jemus42", "ratings")

build_trakt_url("shows", "popular", page = 1, limit = 5)
```

collected_media *Most collected media*

Description

These functions return the most collected movies/shows on trakt.tv.

Usage

```
movies_collected(
  limit = 10,
  extended = "min",
  period = c("weekly", "monthly", "yearly", "all"),
  filters = NULL,
  query = NULL,
  years = NULL,
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL
)

shows_collected(
  limit = 10,
  extended = "min",
  period = c("weekly", "monthly", "yearly", "all"),
  filters = NULL,
  query = NULL,
  years = NULL,
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL,
  networks = NULL,
  status = NULL
)
```

Arguments

limit	`integer(1)` `[10L]`: Number of items to return. Must be greater than `0` and will be coerced via `as.integer()`.
extended	`character`: Level of detail for the API response.

- `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param.
- `"full"`: Complete info including overview, ratings, runtime, etc.
- `"images"`: Minimal info plus image URLs (returned as a list-column).
- `"full,images"`: Complete info plus images.
- `"metadata"`: Collection endpoints only; adds video/audio metadata.

Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

`period` `character(1)` `["weekly"]`: Which period to filter by. Possible values are `"weekly"`, `"monthly"`, `"yearly"`, `"all"`.

`filters` A `trakt_filters` object created with `filters_movies()`, `filters_shows()`, or `filters_episodes()` that refines which items are returned. See filters for the full set of supported filters. Supplying filters as individual arguments (`genres`, `years`, `networks`, ...) is soft-deprecated in favour of this argument; if both are given, `filters` takes precedence.

`query` `character(1)`: Search string for titles and descriptions. For `search_query()` other fields are searched depending on the `type` of media. See the API docs for a full reference.

`years` `character | integer`: 4-digit year (`2010`) **or** range, e.g. `"2010-2020"`. Can also be an integer vector of length two which will be coerced appropriately, e.g. `c(2010, 2020)`.

`genres` `character(n)`: Genre slug(s). See `trakt_genres` for a table of genres. Multiple values are allowed and will be concatenated.

`languages` `character(n)`: Two-letter language code(s). Also see `trakt_languages` for available languages (code and name).

`countries` `character(n)`: Two-letter country code(s). See `trakt_countries`.

`runtimes` `character | integer`: Integer range in minutes, e.g. `30-90`. Can also be an integer vector of length two which will be coerced appropriately.

`ratings` `character | integer`: Integer range between `0` and `100`. Can also be an integer vector of length two which will be coerced appropriately. Note that user-supplied ratings are in the range of 1 to 10, yet the ratings on the site itself are scaled to the range of 1 to 100.

`certifications` `character(n)`: Certification(s) like `pg-13`. Multiple values are allowed. Use `trakt_certifications` for reference. Note that there are different certifications for shows and movies.

`networks` `character(n)`: (Shows only) Network name like `HBO`. See `trakt_networks` for a list of known networks.

`status` `character(n)`: (Shows only) The status of the shows. One of `"returning series"`, `"in production"`, `"planned"`, `"canceled"`, or `"ended"`.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

The Dynamic Lists on trakt.tv

These functions access the automatically updated lists provided by trakt.tv. Each function comes in two flavors: Shows or movies. The following descriptions are adapted directly from the API reference.

- Popular: Popularity is calculated using the rating percentage and the number of ratings.
- Trending: Returns all movies/shows being watched right now. Movies/shows with the most users are returned first.
- Played: Returns the most played (a single user can watch multiple times) movies/shows in the specified time `period`.
- Watched: Returns the most watched (unique users) movies/shows in the specified time `period`.
- Collected: Returns the most collected (unique users) movies/shows in the specified time `period`.
- Anticipated: Returns the most anticipated movies/shows based on the number of lists a movie/show appears on. The functions for **Played**, **Watched**, **Collected** and **Played** each return the same additional variables besides the media information: `watcher_count`, `play_count`, `collected_count`, `collector_count`.

Source

`movies_collected()` wraps endpoint /movies/collected/:period.

`shows_collected()` wraps endpoint /shows/collected/:period.

See Also

Other movie data: `anticipated_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `movies_boxoffice()`, `movies_related()`, `movies_releases()`, `movies_summary()`, `people_media()`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other dynamic lists: `anticipated_media`, `filters`, `lists_popular()`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other show data: `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `people_media()`, `played_media`, `shows_next_episode()`, `shows_related()`, `shows_summary()`, `updated_media`

`comments_comment` *Get a single comment*

Description

Get a single comment

Usage

```
comments_comment(id, extended = "min")

comments_replies(id, extended = "min")

comments_likes(id, extended = "min")

comments_item(id, extended = "min")
```

Arguments

id	character(1): The ID of the item requested. Preferably the trakt ID (e.g. 1429). Other options are the trakt.tv slug (e.g. "the-wire") or imdb ID (e.g. "tt0306414"). Can also be of length greater than 1, in which case the function is called on all id values separately and the result is combined. See vignette("tRakt") for more details.
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Functions

- comments_replies(): Get a comment's replies
- comments_likes(): Get users who liked a comment.
- comments_item(): Get the media item attached to the comment.

Source

comments_comment() wraps endpoint /comments/:id.

comments_replies() wraps endpoint /comments/:id/replies.

comments_likes() wraps endpoint /comments/:id/likes.

comments_item() wraps endpoint /comments/:id/item.

See Also

Other comment methods: comments_trending(), comments_updates(), media_comments, user_comments(), user_list_comments()

Other summary methods: episodes_summary(), movies_summary(), people_summary(), seasons_summary(), shows_summary(), user_profile()

Examples

```
# A single comment
comments_comment("236397")

# Multiple comments
comments_comment(c("236397", "112561"))


comments_replies("236397")


comments_likes("236397")


# A movie
comments_item("236397")
comments_item("236397", extended = "full")

# A show
comments_item("120768")
comments_item("120768", extended = "full")

# A season
comments_item("140265")
comments_item("140265", extended = "full")

# An episode
comments_item("136632")
comments_item("136632", extended = "full")
```

---

comments_trending *Get trending or recently made comments*

Description

Get trending or recently made comments

Usage

```
comments_trending(
  comment_type = c("all", "reviews", "shouts"),
  type = c("all", "movies", "shows", "seasons", "episodes", "lists"),
  include_replies = FALSE,
  limit = 10L
)

comments_recent(
  comment_type = c("all", "reviews", "shouts"),
  type = c("all", "movies", "shows", "seasons", "episodes", "lists"),
  include_replies = FALSE,
  limit = 10L
)
```

Arguments

comment_type	character(1) ["all"]: The type of comment, one of "all", "reviews" or "shouts".
type	character(1) ["all"]: The type of media to filter by, one of "all", "movies", "shows", "seasons", "episodes" or "lists".
include_replies	logical(1) [FALSE]: Whether to include replies.
limit	integer(1) [10L]: Number of items to return. Must be greater than 0 and will be coerced via as.integer().

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Source

comments_trending() wraps endpoint /comments/trending/:comment_type/:type?include_replies=.

comments_recent() wraps endpoint /comments/recent/:comment_type/:type?include_replies=.

See Also

Other comment methods: comments_comment(), comments_updates(), media_comments, user_comments(), user_list_comments()

Examples

```
# Trending reviews
comments_trending("reviews")

# Recent shouts (short comments)
comments_recent("shouts")
```

`comments_updates` *Get recently updated/edited comments*

Description

Get recently updated/edited comments

Usage

```
comments_updates(
  comment_type = c("all", "reviews", "shouts"),
  type = c("all", "movies", "shows", "seasons", "episodes", "lists"),
  include_replies = FALSE,
  limit = 10L
)
```

Arguments

`comment_type`	`character(1)` `["all"]`: The type of comment, one of "all", "reviews" or "shouts".
`type`	`character(1)` `["all"]`: The type of media to filter by, one of "all", "movies", "shows", "seasons", "episodes" or "lists".
`include_replies`	`logical(1)` `[FALSE]`: Whether to include replies.
`limit`	`integer(1)` `[10L]`: Number of items to return. Must be greater than `0` and will be coerced via `as.integer()`.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`comments_updates()` wraps endpoint /comments/updates/:comment_type/:type?include_replies=.

See Also

Other comment methods: `comments_comment()`, `comments_trending()`, `media_comments`, `user_comments()`, `user_list_comments()`

Examples

```
## Not run:  # quite slow
# Recently updated comments
comments_updates(limit = 2)

## End(Not run)
```

`episodes_summary` *Get a single episode's details*

Description

This retrieves a single episode. See `seasons_episodes()` for a whole season, and `seasons_summary()` for (potentially) all episodes of a show.

Usage

```
episodes_summary(id, season = 1L, episode = 1L, extended = "min")
```

Arguments

`id`	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
`season, episode`	`integer(1)` `[1L]`: The season and episode number. If longer, e.g. `1:5`, the function is vectorized and the output will be combined. This may result in *a lot* of API calls. Use wisely.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`episodes_summary()` wraps endpoint /shows/:id/seasons/:season/episodes/:episode.

See Also

Other episode data: media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, seasons_episodes(), seasons_summary(), shows_next_episode()

Other summary methods: comments_comment(), movies_summary(), people_summary(), seasons_summary(), shows_summary(), user_profile()

Examples

```
# Get just this one episode with its ratings, votes, etc.
episodes_summary("breaking-bad", season = 1, episode = 1, extended = "full")
```

---

filters *Build a filter set for the dynamic-list and search endpoints*

Description

The trakt.tv API lets several movies, shows, and search methods refine their results with *filters*. These builders assemble a validated set of filters to pass to the filters argument of functions like movies_popular(), shows_trending(), or search_query().

Usage

```
filters_movies(
  query = NULL,
  years = NULL,
  genres = NULL,
  subgenres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  studio_ids = NULL,
  ratings = NULL,
  votes = NULL,
  tmdb_ratings = NULL,
  tmdb_votes = NULL,
  imdb_ratings = NULL,
  imdb_votes = NULL,
  rt_meters = NULL,
  rt_user_meters = NULL,
  metascores = NULL,
  certifications = NULL
)

filters_shows(
  query = NULL,
```

```
  years = NULL,
  genres = NULL,
  subgenres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  studio_ids = NULL,
  ratings = NULL,
  votes = NULL,
  tmdb_ratings = NULL,
  tmdb_votes = NULL,
  imdb_ratings = NULL,
  imdb_votes = NULL,
  certifications = NULL,
  network_ids = NULL,
  networks = NULL,
  status = NULL
)

filters_episodes(
  query = NULL,
  years = NULL,
  genres = NULL,
  subgenres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  studio_ids = NULL,
  ratings = NULL,
  votes = NULL,
  tmdb_ratings = NULL,
  tmdb_votes = NULL,
  imdb_ratings = NULL,
  imdb_votes = NULL,
  certifications = NULL,
  network_ids = NULL,
  episode_types = NULL
)
```

Arguments

`query`	`character(1)`: Match titles and descriptions.
`years`	`numeric`: A single 4-digit year or a length-2 range, e.g. `2016` or `c(2000, 2010)`.
`genres`, `languages`, `countries`, `certifications`	`character`: One or more genre slugs, language codes, country codes, or certification slugs.
`subgenres`	`character`: One or more subgenre slugs (technical preview).
`runtimes`	`numeric`: Runtime in minutes; a single value or length-2 range.

`studio_ids`, `network_ids`
: `integer`: One or more Trakt studio / network IDs.

`ratings`, `votes`
: `numeric`: Trakt rating (`0`-`100`) / vote-count (`0`-`100000`) range.

`tmdb_ratings`, `tmdb_votes`
: `numeric`: TMDB rating (`0.0`-`10.0`) / vote-count range.

`imdb_ratings`, `imdb_votes`
: `numeric`: IMDb rating (`0.0`-`10.0`) / vote-count range.

`rt_meters`, `rt_user_meters`
: `numeric`: Rotten Tomatoes tomatometer / audience-score range (`0`-`100`). Movies only.

`metascores`
: `numeric`: Metacritic score range (`0`-`100`). Movies only.

`networks`
: `character`: One or more network names (e.g. `"Netflix"`), validated against trakt_networks. For precise matching prefer `network_ids`.

`status`
: `character`: One or more show statuses: `"returning series"`, `"continuing"`, `"in production"`, `"planned"`, `"upcoming"`, `"pilot"`, `"canceled"`, `"ended"`.

`episode_types`
: `character`: One or more episode types: `"standard"`, `"series_premiere"`, `"season_premiere"`, `"mid_season_finale"`, `"mid_season_premiere"`, `"season_finale"`, `"series_finale"`.

Details

Each builder exposes exactly the filters the API supports for that media type. Values are validated and normalized up front (unknown vocabulary values warn and are dropped; out-of-range numbers warn and are ignored), so a malformed filter never silently produces a wrong request.

Value

A `trakt_filters` object: a validated, classed list of query parameters. Pass it to a function's `filters` argument.

See Also

Other dynamic lists: `anticipated_media`, `collected_media`, `lists_popular()`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Examples

```
filters_movies(genres = "action", years = c(2000, 2010), imdb_ratings = c(8, 10))

filters_shows(networks = "Netflix", status = "returning series")

filters_episodes(episode_types = "season_finale")
```

`futurama` *Futurama episodes*

Description

This data comes from `https://trakt.tv` and serves as an example of episode data output.

Usage

```
futurama
```

Format

A tibble() with 124 rows and 18 variables:

episode, season Episode within each season and season number

title Episode title

episode_abs Overall episode number

overview Episode summary

rating Rating (1-10) on trakt.tv

votes Number of votes ontrakt.tv

comment_count Number of comments on episode page

first_aired, updated_at Original air date and last update in UTC as `POSIXct`

runtime Runtime in minutes

trakt, tvdb, tmdb Episode IDs for trakt.tv, TVDb, and TMDb

available_translations List-column of available translation on trakt.tv

See Also

Other Episode datasets: `gameofthrones`

Examples

```
futurama
```

gameofthrones *Game of Thrones episodes*

Description

This data comes from https://trakt.tv and https://en.wikipedia.org/wiki/List_of_Game_of_Thrones_episodes.

Usage

```
gameofthrones
```

Format

A tibble() with 67 rows and 17 variables:

episode_abs Overall episode number

episode, season Episode within each season and season number

title Episode title

overview Episode summary

rating Rating (1-10) on trakt.tv

votes Number of votes ontrakt.tv

viewers Viewers according to Wikipedia

director, writer Directing and writing credits

comment_count Number of comments on episode page

first_aired, updated_at Original air date and last update in UTC as `POSIXct`

runtime Runtime in minutes

trakt, tvdb, tmdb Episode IDs for trakt.tv, TVDb, and TMDb

year Year of first airing

epid Episode ID in `s00e00` format

See Also

Other Episode datasets: futurama

Examples

```
gameofthrones
```

`lists_popular` *Get popular / trending lists*

Description

Get popular / trending lists

Usage

```
lists_popular(limit = 10, type = c("personal", "official"))

lists_trending(limit = 10, type = c("personal", "official"))
```

Arguments

`limit`	`integer(1)` `[10L]`: Number of items to return. Must be greater than `0` and will be coerced via `as.integer()`.
`type`	`character(1)` `["personal"]`: The kind of lists to return, one of `"personal"` (user-created lists) or `"official"` (Trakt-curated lists). The trakt.tv API requires this path segment; a request without it returns an empty (HTTP 204) response.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`lists_popular()` wraps endpoint /lists/popular/:type.

`lists_trending()` wraps endpoint /lists/trending/:type.

See Also

`user_list_items()` For the actual content of a list.

Other list methods: `media_lists`, `user_list()`, `user_list_comments()`, `user_list_items()`, `user_lists()`

Other dynamic lists: `anticipated_media`, `collected_media`, `filters`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Examples

```
lists_popular()
lists_trending()
```

media_aliases *Get all movie / show aliases*

Description

Get all movie / show aliases

Usage

```
movies_aliases(id)

shows_aliases(id)
```

Arguments

id	character(1): The ID of the item requested. Preferably the trakt ID (e.g. 1429). Other options are the trakt.tv slug (e.g. "the-wire") or imdb ID (e.g. "tt0306414"). Can also be of length greater than 1, in which case the function is called on all id values separately and the result is combined. See vignette("tRakt") for more details.

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Source

movies_aliases() wraps endpoint /movies/:id/aliases.

shows_aliases() wraps endpoint /shows/:id/aliases.

See Also

Other movie data: anticipated_media, collected_media, media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, movies_boxoffice(), movies_related(), movies_releases(), movies_summary(), people_media(), played_media, popular_media, trending_media, updated_media, watched_media

Other show data: collected_media, media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, people_media(), played_media, shows_next_episode(), shows_related(), shows_summary(), updated_media

Examples

```
movies_aliases(190430)
shows_aliases(104439)
```

media_comments *Get all comments of a thing*

Description

Get all comments of a thing

Usage

```
movies_comments(
  id,
  sort = c("newest", "oldest", "likes", "replies"),
  extended = "min",
  limit = 10L
)

shows_comments(
  id,
  sort = c("newest", "oldest", "likes", "replies"),
  extended = "min",
  limit = 10L
)

seasons_comments(
  id,
  season = 1L,
  sort = c("newest", "oldest", "likes", "replies"),
  extended = "min",
  limit = 10L
)

episodes_comments(
  id,
  season = 1L,
  episode = 1L,
  sort = c("newest", "oldest", "likes", "replies"),
  extended = "min",
  limit = 10L
)
```

Arguments

id	character(1): The ID of the item requested. Preferably the trakt ID (e.g. 1429). Other options are the trakt.tv slug (e.g. "the-wire") or imdb ID (e.g. "tt0306414"). Can also be of length greater than 1, in which case the function is called on all id values separately and the result is combined. See vignette("tRakt") for more details.

`sort` `character(1)` `["newest"]`: Comment sort order, one of "newest", "oldest", "likes" or "replies".

`extended` `character`: Level of detail for the API response.

- `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param.
- `"full"`: Complete info including overview, ratings, runtime, etc.
- `"images"`: Minimal info plus image URLs (returned as a list-column).
- `"full,images"`: Complete info plus images.
- `"metadata"`: Collection endpoints only; adds video/audio metadata.

Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

`limit` `integer(1)` `[10L]`: Number of items to return. Must be greater than `0` and will be coerced via `as.integer()`.

`season, episode` `integer(1)` `[1L]`: The season and episode number. If longer, e.g. `1:5`, the function is vectorized and the output will be combined. This may result in *a lot* of API calls. Use wisely.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Functions

- `movies_comments()`: Get comments for a movie
- `shows_comments()`: Get comments for a movie
- `seasons_comments()`: Get comments for a season
- `episodes_comments()`: Get comments for an episode

Source

`movies_comments()` wraps endpoint /movies/:id/comments/:sort.

`shows_comments()` wraps endpoint /shows/:id/comments/:sort.

`seasons_comments()` wraps endpoint /shows/:id/seasons/:season/comments/:sort.

`episodes_comments()` wraps endpoint /shows/:id/seasons/:season/episodes/:episode/comments/:sort.

See Also

Other comment methods: `comments_comment()`, `comments_trending()`, `comments_updates()`, `user_comments()`, `user_list_comments()`

Other movie data: `anticipated_media`, `collected_media`, `media_aliases`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `movies_boxoffice()`, `movies_related()`, `movies_releases()`, `movies_summary()`, `people_media()`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other show data: collected_media, media_aliases, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, people_media(), played_media, shows_next_episode(), shows_related(), shows_summary(), updated_media

Other season data: media_lists, media_people, media_ratings(), media_stats(), seasons_episodes(), seasons_season(), seasons_summary()

Other episode data: episodes_summary(), media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, seasons_episodes(), seasons_summary(), shows_next_episode()

Examples

```
movies_comments(193972)
shows_comments(46241, sort = "likes")
seasons_comments(46241, season = 1, sort = "likes")
episodes_comments(46241, season = 1, episode = 2, sort = "likes")
```

---

media_lists *Get lists containing a movie, show, season, episode or person*

Description

Get lists containing a movie, show, season, episode or person

Usage

```
movies_lists(
  id,
  type = c("all", "personal", "official", "watchlists"),
  sort = c("popular", "likes", "comments", "items", "added", "updated"),
  limit = 10L,
  extended = "min"
)

shows_lists(
  id,
  type = c("all", "personal", "official", "watchlists"),
  sort = c("popular", "likes", "comments", "items", "added", "updated"),
  limit = 10L,
  extended = "min"
)

seasons_lists(
  id,
  season,
  type = c("all", "personal", "official", "watchlists"),
  sort = c("popular", "likes", "comments", "items", "added", "updated"),
  limit = 10L,
```

```
  extended = "min"
)

episodes_lists(
  id,
  season,
  episode,
  type = c("all", "personal", "official", "watchlists"),
  sort = c("popular", "likes", "comments", "items", "added", "updated"),
  limit = 10L,
  extended = "min"
)

people_lists(
  id,
  type = c("all", "personal", "official", "watchlists"),
  sort = c("popular", "likes", "comments", "items", "added", "updated"),
  limit = 10L,
  extended = "min"
)
```

Arguments

id	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
type	`character(1)` `["all"]`: The type of list, one of "all", "personal", "official" or "watchlists".
sort	`character(1)` `["popular"]`: Sort lists by one of "popular", "likes", "comments", "items", "added" or "updated".
limit	`integer(1)` `[10L]`: Number of items to return. Must be greater than `0` and will be coerced via `as.integer()`.
extended	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).
season, episode	`integer(1)` `[1L]`: The season and episode number. If longer, e.g. `1:5`, the function is vectorized and the output will be combined. This may result in *a lot* of API calls. Use wisely.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Functions

- `movies_lists()`: Lists containing a movie.
- `shows_lists()`: Lists containing a show.
- `seasons_lists()`: Lists containing a season.
- `episodes_lists()`: Lists containing an episode.
- `people_lists()`: Lists containing a person.

Source

`movies_lists()` wraps endpoint /movies/:id/lists/:type/:sort.

`shows_lists()` wraps endpoint /shows/:id/lists/:type/:sort.

`seasons_lists()` wraps endpoint /shows/:id/seasons/:season/lists/:type/:sort.

`episodes_lists()` wraps endpoint /shows/:id/seasons/:season/episodes/:episode/lists/:type/:sort.

See Also

Other list methods: `lists_popular()`, `user_list()`, `user_list_comments()`, `user_list_items()`, `user_lists()`

Other movie data: `anticipated_media`, `collected_media`, `media_aliases`, `media_comments`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `movies_boxoffice()`, `movies_related()`, `movies_releases()`, `movies_summary()`, `people_media()`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other show data: `collected_media`, `media_aliases`, `media_comments`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `people_media()`, `played_media`, `shows_next_episode()`, `shows_related()`, `shows_summary()`, `updated_media`

Other season data: `media_comments`, `media_people`, `media_ratings()`, `media_stats()`, `seasons_episodes()`, `seasons_season()`, `seasons_summary()`

Other episode data: `episodes_summary()`, `media_comments`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `seasons_episodes()`, `seasons_summary()`, `shows_next_episode()`

Other people data: `media_people`, `people_media()`, `people_summary()`

Examples

```
movies_lists("190430", type = "personal", limit = 5)


shows_lists("46241")
```

```
seasons_lists("46241", season = 1)


episodes_lists("46241", season = 1, episode = 1)


people_lists("david-tennant")

people_lists("emilia-clarke", sort = "items")
```

`media_people` *Get the cast and crew of a show or movie*

Description

Returns all cast and crew for a show/movie, depending on how much data is available.

Usage

```
movies_people(id, extended = "min")

shows_people(id, guest_stars = FALSE, extended = "min")

seasons_people(id, season = 1L, guest_stars = FALSE, extended = "min")

episodes_people(
  id,
  season = 1L,
  episode = 1L,
  guest_stars = FALSE,
  extended = "min"
)
```

Arguments

`id`	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column).

- `"full,images"`: Complete info plus images.
- `"metadata"`: Collection endpoints only; adds video/audio metadata.

Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

`guest_stars` **[Deprecated]** `logical(1)` `["FALSE"]`: Previously requested a separate `guest_stars` table. The trakt.tv API no longer returns that array — guest cast is now included in `cast`. This argument is currently a no-op and will be removed in a future release.

`season, episode` `integer(1)` `[1L]`: The season and episode number. If longer, e.g. `1:5`, the function is vectorized and the output will be combined. This may result in *a lot* of API calls. Use wisely.

Value

A `list` of one or more tibbles for `cast` and/or `crew`. The latter `tibble` objects are as flat as possible.

Note

As of 2019-09-30, there are two representations of `character[s]` and `job[s]`: One is a regular character variable, and the other is a list-column. The former is deprecated and only included for compatibility reasons.

Source

`movies_people()` wraps endpoint /movies/:id/people.

`shows_people()` wraps endpoint /shows/:id/people.

`seasons_people()` wraps endpoint /shows/:id/seasons/:season/people.

`episodes_people()` wraps endpoint /shows/:id/seasons/:season/episodes/:episode/people.

See Also

people_media, for the other direction: People that have credits in shows/movies.

Other people data: `media_lists`, `people_media()`, `people_summary()`

Other movie data: `anticipated_media`, `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `movies_boxoffice()`, `movies_related()`, `movies_releases()`, `movies_summary()`, `people_media()`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other show data: `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `people_media()`, `played_media`, `shows_next_episode()`, `shows_related()`, `shows_summary()`, `updated_media`

Other season data: `media_comments`, `media_lists`, `media_ratings()`, `media_stats()`, `seasons_episodes()`, `seasons_season()`, `seasons_summary()`

Other episode data: `episodes_summary()`, `media_comments`, `media_lists`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `seasons_episodes()`, `seasons_summary()`, `shows_next_episode()`

Examples

```
movies_people("deadpool-2016")
shows_people("breaking-bad")
seasons_people("breaking-bad", season = 1)
episodes_people("breaking-bad", season = 1, episode = 1)
```

`media_ratings` *Media user ratings*

Description

Returns a movie's or show's (or season's, or episode's) rating and ratings distribution. If you *do not* want the full ratings distribution, it is highly advised to just use `*_summary` functions or seasons_episodes for episode ratings.

Usage

```
shows_ratings(id)

movies_ratings(id)

seasons_ratings(id, season = 1L)

episodes_ratings(id, season = 1L, episode = 1L)
```

Arguments

`id`	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
`season, episode`	`integer(1) [1L]`: The season and episode number. If longer, e.g. `1:5`, the function is vectorized and the output will be combined. This may result in *a lot* of API calls. Use wisely.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Note

Since this function is able to work on multi-length inputs for `id`, `season` and `episode`, it is possible to get a lot of data, *but* at the cost of one API call *per element in each argument*. Please be kind to the API.

Source

shows_ratings() wraps endpoint /shows/:id/ratings.

movies_ratings() wraps endpoint /movies/:id/ratings.

seasons_ratings() wraps endpoint /shows/:id/seasons/:season/ratings.

episodes_ratings() wraps endpoint /shows/:id/seasons/:season/episodes/:episode/ratings.

See Also

Other show data: collected_media, media_aliases, media_comments, media_lists, media_people, media_stats(), media_translations, media_watching, people_media(), played_media, shows_next_episode(), shows_related(), shows_summary(), updated_media

Other movie data: anticipated_media, collected_media, media_aliases, media_comments, media_lists, media_people, media_stats(), media_translations, media_watching, movies_boxoffice(), movies_related(), movies_releases(), movies_summary(), people_media(), played_media, popular_media, trending_media, updated_media, watched_media

Other season data: media_comments, media_lists, media_people, media_stats(), seasons_episodes(), seasons_season(), seasons_summary()

Other episode data: episodes_summary(), media_comments, media_lists, media_people, media_stats(), media_translations, media_watching, seasons_episodes(), seasons_summary(), shows_next_episode()

Examples

```
# A movie's ratings
movies_ratings("tron-legacy-2010")

# A show's ratings
shows_ratings("game-of-thrones")
## Not run:
# Ratings for seasons 1 through 5
seasons_ratings("futurama", season = 1:5)

# Ratings for episodes 1 through 7 of season 1
episodes_ratings("futurama", season = 1, episode = 1:7)

## End(Not run)
```

---

media_stats *Get a show or movie's (or season's or episode's) stats*

Description

The data contains watchers, playes, collectors, comments, lists, and votes.

Usage

```
shows_stats(id)

movies_stats(id)

seasons_stats(id, season = 1L)

episodes_stats(id, season = 1L, episode = 1L)
```

Arguments

id	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
season, episode	`integer(1)` `[1L]`: The season and episode number. If longer, e.g. `1:5`, the function is vectorized and the output will be combined. This may result in *a lot* of API calls. Use wisely.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`shows_stats()` wraps endpoint /shows/:id/stats.

`movies_stats()` wraps endpoint /movies/:id/stats.

`seasons_stats()` wraps endpoint /shows/:id/seasons/:season/stats.

`episodes_stats()` wraps endpoint /shows/:id/seasons/:season/episodes/:episode/stats.

See Also

Other show data: `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_translations`, `media_watching`, `people_media()`, `played_media`, `shows_next_episode()`, `shows_related()`, `shows_summary()`, `updated_media`

Other movie data: `anticipated_media`, `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_translations`, `media_watching`, `movies_boxoffice()`, `movies_related()`, `movies_releases()`, `movies_summary()`, `people_media()`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other season data: `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `seasons_episodes()`, `seasons_season()`, `seasons_summary()`

Other episode data: `episodes_summary()`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_translations`, `media_watching`, `seasons_episodes()`, `seasons_summary()`, `shows_next_episode()`

Examples

```
# Stats for a movie
movies_stats("inception-2010")
## Not run:
# Stats for multiple shows at once
shows_stats(c("breaking-bad", "game-of-thrones"))

# Stats for multiple episodes
episodes_stats("futurama", season = 1, episode = 1:7)

## End(Not run)
```

`media_translations` *Get translations for a movie, show or episode*

Description

Get translations for a movie, show or episode

Usage

```
movies_translations(id, languages = NULL)

shows_translations(id, languages = NULL)

episodes_translations(id, season = 1L, episode = 1L, languages = NULL)
```

Arguments

`id`	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
`languages`	`character(n)`: Two-letter language code(s). Also see `trakt_languages` for available languages (code and name).
`season, episode`	`integer(1)` `[1L]`: The season and episode number. If longer, e.g. `1:5`, the function is vectorized and the output will be combined. This may result in *a lot* of API calls. Use wisely.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`movies_translations()` wraps endpoint /movies/:id/translations/:language.

`shows_translations()` wraps endpoint /shows/:id/translations/:language.

`episodes_translations()` wraps endpoint /shows/:id/seasons/:season/episodes/:episode/translations/:language.

See Also

Other movie data: `anticipated_media`, `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_watching`, `movies_boxoffice()`, `movies_related()`, `movies_releases()`, `movies_summary()`, `people_media()`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other show data: `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_watching`, `people_media()`, `played_media`, `shows_next_episode()`, `shows_related()`, `shows_summary()`, `updated_media`

Other episode data: `episodes_summary()`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_watching`, `seasons_episodes()`, `seasons_summary()`, `shows_next_episode()`

Examples

```
# Get all translations
movies_translations("193972")

# Only get a specific language
movies_translations("193972", "de")
```

---

media_watching *Get who's watching a thing right now*

Description

Get who's watching a thing right now

Usage

```
movies_watching(id, extended = "min")

shows_watching(id, extended = "min")

seasons_watching(id, season = 1L, extended = "min")

episodes_watching(id, season = 1L, episode = 1L, extended = "min")
```

Arguments

id	character(1): The ID of the item requested. Preferably the trakt ID (e.g. 1429). Other options are the trakt.tv slug (e.g. "the-wire") or imdb ID (e.g. "tt0306414"). Can also be of length greater than 1, in which case the function is called on all id values separately and the result is combined. See vignette("tRakt") for more details.
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).
season, episode	integer(1) [1L]: The season and episode number. If longer, e.g. 1:5, the function is vectorized and the output will be combined. This may result in *a lot* of API calls. Use wisely.

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Functions

- movies_watching(): Who's watching a movie.
- shows_watching(): Who's watching a show.
- seasons_watching(): Who's watching a season.
- episodes_watching(): Who's watching an episode.

Source

movies_watching() wraps endpoint /movies/:id/watching.

shows_watching() wraps endpoint /shows/:id/watching.

episodes_watching() wraps endpoint /shows/:id/seasons/:season/episodes/:episode/watching.

See Also

Other movie data: anticipated_media, collected_media, media_aliases, media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, movies_boxoffice(), movies_related(), movies_releases(), movies_summary(), people_media(), played_media, popular_media, trending_media, updated_media, watched_media

Other show data: collected_media, media_aliases, media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, people_media(), played_media, shows_next_episode(), shows_related(), shows_summary(), updated_media

Other episode data: episodes_summary(), media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, seasons_episodes(), seasons_summary(), shows_next_episode()

Examples

```
movies_watching("deadpool-2016")
shows_watching("the-simpsons")
seasons_watching("the-simpsons", season = 6)
episodes_watching("the-simpsons", season = 6, episode = 12)
```

---

movies_boxoffice *Get the weekend box office*

Description

Returns the top 10 grossing movies in the U.S. box office last weekend. Updated every Monday morning.

Usage

```
movies_boxoffice(extended = "min")
```

Arguments

extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Source

movies_boxoffice() wraps endpoint /movies/boxoffice.

See Also

Other movie data: anticipated_media, collected_media, media_aliases, media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, movies_related(), movies_releases(), movies_summary(), people_media(), played_media, popular_media, trending_media, updated_media, watched_media

Examples

```
movies_boxoffice()
```

---

movies_related *Get similiar(ish) movies*

Description

Get similiar(ish) movies

Usage

```
movies_related(id, limit = 10L, extended = "min")
```

Arguments

id	character(1): The ID of the item requested. Preferably the trakt ID (e.g. 1429). Other options are the trakt.tv slug (e.g. "the-wire") or imdb ID (e.g. "tt0306414"). Can also be of length greater than 1, in which case the function is called on all id values separately and the result is combined. See vignette("tRakt") for more details.
limit	integer(1) [10L]: Number of items to return. Must be greater than 0 and will be coerced via as.integer().
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Source

movies_related() wraps endpoint /movies/:id/related.

See Also

Other movie data: anticipated_media, collected_media, media_aliases, media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, movies_boxoffice(), movies_releases(), movies_summary(), people_media(), played_media, popular_media, trending_media, updated_media, watched_media

Examples

```
movies_related("the-avengers-2012", limit = 5)
```

---

movies_releases *Get a movie's release details*

Description

Retrieve one or more movie's release information, including the release date, country code (two letter, e.g. us), and the certification (e.g. PG).

Usage

```
movies_releases(id, country = NULL)
```

Arguments

id	character(1): The ID of the item requested. Preferably the trakt ID (e.g. 1429). Other options are the trakt.tv slug (e.g. "the-wire") or imdb ID (e.g. "tt0306414"). Can also be of length greater than 1, in which case the function is called on all id values separately and the result is combined. See vignette("tRakt") for more details.
country	Optional two letter country code to filter by. See trakt_countries for a table of country codes.

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Source

movies_releases() wraps endpoint /movies/:id/releases/:country.

See Also

Other movie data: anticipated_media, collected_media, media_aliases, media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, movies_boxoffice(), movies_related(), movies_summary(), people_media(), played_media, popular_media, trending_media, updated_media, watched_media

Examples

```
movies_releases("tron-legacy-2010")
```

`movies_summary` *Get a single movie*

Description

Get a single movie

Usage

```
movies_summary(id, extended = "min")
```

Arguments

`id`	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

movies_summary() wraps endpoint /movies/:id.

See Also

Other movie data: anticipated_media, collected_media, media_aliases, media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, movies_boxoffice(), movies_related(), movies_releases(), people_media(), played_media, popular_media, trending_media, updated_media, watched_media

Other summary methods: comments_comment(), episodes_summary(), people_summary(), seasons_summary(), shows_summary(), user_profile()

Examples

```
# Minimal info by default
movies_summary("inception-2010")
## Not run:
# Full information,  multiple movies
movies_summary(c("inception-2010", "the-dark-knight-2008"), extended = "full")

## End(Not run)
```

---

pad_episode *Easy episode number padding*

Description

Simple function to ease the creation of sXXeYY episode ids. Note that s and e must have the same length.

Usage

```
pad_episode(s = "0", e = "0", s_width = 2, e_width = 2)
```

Arguments

s	Input season number, coerced to character.
e	Input episode number, coerced to character.
s_width	The length of the season number padding. Defaults to 2.
e_width	The length of the episode number padding. Defaults to 2.

Value

A character in the common sXXeYY format

Note

I like my sXXeYY format, okay?

See Also

Other utility functions: `build_trakt_url()`

Examples

```
# Season 2, episode 4
pad_episode(2, 4)
pad_episode(1, 85, e_width = 3)
```

`people_media` *Get a single person's movie or show credits*

Description

Returns all movies or shows where this person is in the cast or crew.

Usage

```
people_movies(id, extended = "min")

people_shows(id, extended = "min")
```

Arguments

`id`	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Details

Note that as of 2019-09-30, there are two representations of `character[s]` and `job[s]`: One is a regular character variable, and the other is a list-column. The singular is deprecated and only included for compatibility reasons.

Value

A `list` of one or more tibbles for `cast` and `crew`. The latter `tibble` objects are as flat as possible.

Source

`people_movies()` wraps endpoint /people/:id/movies.

`people_shows()` wraps endpoint /people/:id/shows.

See Also

media_people, for the other direction: Media that has people.

Other movie data: `anticipated_media`, `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `movies_boxoffice()`, `movies_related()`, `movies_releases()`, `movies_summary()`, `played_media`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other people data: `media_lists`, `media_people`, `people_summary()`

Other show data: `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `played_media`, `shows_next_episode()`, `shows_related()`, `shows_summary()`, `updated_media`

Examples

```
people_movies("christopher-nolan")

people_shows("kit-harington")
```

---

people_summary *Get a single person's details*

Description

Get a single person's details, like their various IDs. If `extended` is `"full"`, there will also be biographical data if available, e.g. their birthday.

Usage

```
people_summary(id, extended = "min")
```

Arguments

id	character(1): The ID of the item requested. Preferably the trakt ID (e.g. 1429). Other options are the trakt.tv slug (e.g. "the-wire") or imdb ID (e.g. "tt0306414"). Can also be of length greater than 1, in which case the function is called on all id values separately and the result is combined. See vignette("tRakt") for more details.
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Source

people_summary() wraps endpoint /people/:id.

See Also

Other people data: media_lists, media_people, people_media()

Other summary methods: comments_comment(), episodes_summary(), movies_summary(), seasons_summary(), shows_summary(), user_profile()

Examples

```
# A single person's extended information
people_summary("bryan-cranston", "full")

# Multiple people
people_summary(c("kit-harington", "emilia-clarke"))
```

played_media *Most played media*

Description

These functions return the most played movies/shows on trakt.tv.

Usage

```
movies_played(
  limit = 10,
  extended = "min",
  period = c("weekly", "monthly", "yearly", "all"),
  filters = NULL,
  query = NULL,
  years = NULL,
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL
)

shows_played(
  limit = 10,
  extended = "min",
  period = c("weekly", "monthly", "yearly", "all"),
  filters = NULL,
  query = NULL,
  years = NULL,
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL,
  networks = NULL,
  status = NULL
)
```

Arguments

limit	`integer(1)` `[10L]`: Number of items to return. Must be greater than `0` and will be coerced via `as.integer()`.
extended	`character`: Level of detail for the API response.

- `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param.
- `"full"`: Complete info including overview, ratings, runtime, etc.
- `"images"`: Minimal info plus image URLs (returned as a list-column).
- `"full,images"`: Complete info plus images.
- `"metadata"`: Collection endpoints only; adds video/audio metadata.

Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

`period` `character(1)` `["weekly"]`: Which period to filter by. Possible values are `"weekly"`, `"monthly"`, `"yearly"`, `"all"`.

`filters` A `trakt_filters` object created with `filters_movies()`, `filters_shows()`, or `filters_episodes()` that refines which items are returned. See filters for the full set of supported filters. Supplying filters as individual arguments (`genres`, `years`, `networks`, ...) is soft-deprecated in favour of this argument; if both are given, `filters` takes precedence.

`query` `character(1)`: Search string for titles and descriptions. For `search_query()` other fields are searched depending on the `type` of media. See the API docs for a full reference.

`years` `character | integer`: 4-digit year (`2010`) **or** range, e.g. `"2010-2020"`. Can also be an integer vector of length two which will be coerced appropriately, e.g. `c(2010, 2020)`.

`genres` `character(n)`: Genre slug(s). See `trakt_genres` for a table of genres. Multiple values are allowed and will be concatenated.

`languages` `character(n)`: Two-letter language code(s). Also see `trakt_languages` for available languages (code and name).

`countries` `character(n)`: Two-letter country code(s). See `trakt_countries`.

`runtimes` `character | integer`: Integer range in minutes, e.g. `30-90`. Can also be an integer vector of length two which will be coerced appropriately.

`ratings` `character | integer`: Integer range between `0` and `100`. Can also be an integer vector of length two which will be coerced appropriately. Note that user-supplied ratings are in the range of 1 to 10, yet the ratings on the site itself are scaled to the range of 1 to 100.

`certifications` `character(n)`: Certification(s) like `pg-13`. Multiple values are allowed. Use `trakt_certifications` for reference. Note that there are different certifications for shows and movies.

`networks` `character(n)`: (Shows only) Network name like `HBO`. See `trakt_networks` for a list of known networks.

`status` `character(n)`: (Shows only) The status of the shows. One of `"returning series"`, `"in production"`, `"planned"`, `"canceled"`, or `"ended"`.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

The Dynamic Lists on trakt.tv

These functions access the automatically updated lists provided by trakt.tv. Each function comes in two flavors: Shows or movies. The following descriptions are adapted directly from the API reference.

- Popular: Popularity is calculated using the rating percentage and the number of ratings.
- Trending: Returns all movies/shows being watched right now. Movies/shows with the most users are returned first.
- Played: Returns the most played (a single user can watch multiple times) movies/shows in the specified time `period`.
- Watched: Returns the most watched (unique users) movies/shows in the specified time `period`.
- Collected: Returns the most collected (unique users) movies/shows in the specified time `period`.
- Anticipated: Returns the most anticipated movies/shows based on the number of lists a movie/show appears on. The functions for **Played**, **Watched**, **Collected** and **Played** each return the same additional variables besides the media information: `watcher_count`, `play_count`, `collected_count`, `collector_count`.

Source

`movies_played()` wraps endpoint /movies/played/:period.

`shows_played()` wraps endpoint /shows/played/:period.

See Also

Other movie data: `anticipated_media`, `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `movies_boxoffice()`, `movies_related()`, `movies_releases()`, `movies_summary()`, `people_media()`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other dynamic lists: `anticipated_media`, `collected_media`, `filters`, `lists_popular()`, `popular_media`, `trending_media`, `updated_media`, `watched_media`

Other show data: `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `people_media()`, `shows_next_episode()`, `shows_related()`, `shows_summary()`, `updated_media`

`popular_media` *Popular media*

Description

These functions return the popular movies/shows on trakt.tv.

Usage

```
movies_popular(
  limit = 10,
  extended = "min",
  filters = NULL,
  query = NULL,
  years = NULL,
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL
)

shows_popular(
  limit = 10,
  extended = "min",
  filters = NULL,
  query = NULL,
  years = NULL,
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL,
  networks = NULL,
  status = NULL
)
```

Arguments

`limit`	`integer(1)` `[10L]`: Number of items to return. Must be greater than `0` and will be coerced via `as.integer()`.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).
`filters`	A `trakt_filters` object created with `filters_movies()`, `filters_shows()`, or `filters_episodes()` that refines which items are returned. See filters for the

full set of supported filters. Supplying filters as individual arguments (genres, years, networks, ...) is soft-deprecated in favour of this argument; if both are given, filters takes precedence.

query — character(1): Search string for titles and descriptions. For search_query() other fields are searched depending on the type of media. See the API docs for a full reference.

years — character | integer: 4-digit year (2010) **or** range, e.g. "2010-2020". Can also be an integer vector of length two which will be coerced appropriately, e.g. c(2010, 2020).

genres — character(n): Genre slug(s). See trakt_genres for a table of genres. Multiple values are allowed and will be concatenated.

languages — character(n): Two-letter language code(s). Also see trakt_languages for available languages (code and name).

countries — character(n): Two-letter country code(s). See trakt_countries.

runtimes — character | integer: Integer range in minutes, e.g. 30-90. Can also be an integer vector of length two which will be coerced appropriately.

ratings — character | integer: Integer range between 0 and 100. Can also be an integer vector of length two which will be coerced appropriately. Note that user-supplied ratings are in the range of 1 to 10, yet the ratings on the site itself are scaled to the range of 1 to 100.

certifications — character(n): Certification(s) like pg-13. Multiple values are allowed. Use trakt_certifications for reference. Note that there are different certifications for shows and movies.

networks — character(n): (Shows only) Network name like HBO. See trakt_networks for a list of known networks.

status — character(n): (Shows only) The status of the shows. One of "returning series", "in production", "planned", "canceled", or "ended".

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

The Dynamic Lists on trakt.tv

These functions access the automatically updated lists provided by trakt.tv. Each function comes in two flavors: Shows or movies. The following descriptions are adapted directly from the API reference.

- Popular: Popularity is calculated using the rating percentage and the number of ratings.
- Trending: Returns all movies/shows being watched right now. Movies/shows with the most users are returned first.
- Played: Returns the most played (a single user can watch multiple times) movies/shows in the specified time period.

- Watched: Returns the most watched (unique users) movies/shows in the specified time `period`.
- Collected: Returns the most collected (unique users) movies/shows in the specified time `period`.
- Anticipated: Returns the most anticipated movies/shows based on the number of lists a movie/show appears on. The functions for **Played**, **Watched**, **Collected** and **Played** each return the same additional variables besides the media information: `watcher_count`, `play_count`, `collected_count`, `collector_count`.

Source

`movies_popular()` wraps endpoint /movies/popular.

`shows_popular()` wraps endpoint /shows/popular.

See Also

Other movie data: `anticipated_media`, `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `movies_boxoffice()`, `movies_related()`, `movies_releases()`, `movies_summary()`, `people_media()`, `played_media`, `trending_media`, `updated_media`, `watched_media`

Other dynamic lists: `anticipated_media`, `collected_media`, `filters`, `lists_popular()`, `played_media`, `trending_media`, `updated_media`, `watched_media`

Other shows data: `anticipated_media`, `trending_media`, `watched_media`

Examples

```
# Get the most popular German-language movies between 2000 and 2010
movies_popular(filters = filters_movies(languages = "de", years = c(2000, 2010)))
```

---

search_query *Search trakt.tv via text query or ID*

Description

Search for a show or movie with a keyword (e.g. `"Breaking Bad"`) and receive basic info of the first search result. It's main use is to retrieve the IDs or proper show/movie title for further use, as well as receiving a quick overview of a show/movie.

Usage

```
search_query(
  query,
  type = "show",
  n_results = 1L,
  extended = "min",
  filters = NULL,
```

```
  years = NULL,
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL,
  networks = NULL,
  status = NULL
)

search_id(
  id,
  id_type = c("trakt", "imdb", "tmdb", "tvdb"),
  type = "show",
  n_results = 1L,
  extended = "min"
)
```

Arguments

query	`character(1)`: Search string for titles and descriptions. For `search_query()` other fields are searched depending on the `type` of media. See the API docs for a full reference.
type	`character(1)` `["show"]`: The type of data you're looking for. One of `show`, `movie`, `episode`, `person` or `list` or a character vector with those elements, e.g. `c("show", "movie")`. Note that not every combination is reasonably combinable, e.g. `c("movie", "list")`. Use separate function calls in that case.
n_results	`integer(1)` `[1]`: How many results to return.
extended	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).
filters	A `trakt_filters` object created with `filters_movies()`, `filters_shows()`, or `filters_episodes()` that refines which items are returned. See filters for the full set of supported filters. Supplying filters as individual arguments (`genres`, `years`, `networks`, ...) is soft-deprecated in favour of this argument; if both are given, `filters` takes precedence.
years	`character | integer`: 4-digit year (`2010`) **or** range, e.g. `"2010-2020"`. Can also be an integer vector of length two which will be coerced appropriately, e.g. `c(2010, 2020)`.

genres	character(n): Genre slug(s). See trakt_genres for a table of genres. Multiple values are allowed and will be concatenated.
languages	character(n): Two-letter language code(s). Also see trakt_languages for available languages (code and name).
countries	character(n): Two-letter country code(s). See trakt_countries.
runtimes	character | integer: Integer range in minutes, e.g. 30-90. Can also be an integer vector of length two which will be coerced appropriately.
ratings	character | integer: Integer range between 0 and 100. Can also be an integer vector of length two which will be coerced appropriately. Note that user-supplied ratings are in the range of 1 to 10, yet the ratings on the site itself are scaled to the range of 1 to 100.
certifications	character(n): Certification(s) like pg-13. Multiple values are allowed. Use trakt_certifications for reference. Note that there are different certifications for shows and movies.
networks	character(n): (Shows only) Network name like HBO. See trakt_networks for a list of known networks.
status	character(n): (Shows only) The status of the shows. One of "returning series", "in production", "planned", "canceled", or "ended".
id	character(1): The id used for the search, e.g. 14701 for a Trakt ID.
id_type	character(1) ["trakt"]: The type of id. One of trakt, imdb, tmdb, tvdb.

Details

The amount of information returned is equal to *_summary API methods and in turn depends on the value of extended. See also the API reference here for which fields of the item metadata are searched by default.

Value

A tibble containing n_results results. Variable type is equivalent to the value of the type argument, and variable score indicates the search match, where 1000 is a perfect match. If no results are found, the tibble has 0 rows. If more than one type is specified, e.g. c("movie", "show"), there will be n_results results *per type*.

Source

search_query() wraps endpoint /search/:type?query=.

search_id() wraps endpoint /search/:id_type/:id?type=.

Examples

```
# A show
search_query("Breaking Bad", type = "show", n_results = 3)
## Not run:
# A show by its trakt id, and now with more information
search_id(1388, "trakt", type = "show", extended = "full")
```

```
# A person
search_query("J. K. Simmons", type = "person", extended = "full")

# A movie or a show, two of each
search_query("Tron", type = c("movie", "show"), n_results = 2)

## End(Not run)
```

`seasons_episodes` *Get a season of a show*

Description

Similar to seasons_summary, but this function returns full data for a single season, i.e. all the episodes of the season

Usage

```
seasons_episodes(id, seasons = 1L, extended = "min")
```

Arguments

`id`	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
`seasons`	`integer(1)` `[1L]`: The season(s) to get. Use `0` for specials.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Note

If you want to quickly gather episode data of all available seasons, see seasons_summary and use the `episodes = TRUE` parameter.

Source

`seasons_episodes()` wraps endpoint /shows/:id/seasons/:season?translations=.

See Also

Other season data: `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `seasons_season()`, `seasons_summary()`

Other episode data: `episodes_summary()`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `seasons_summary()`, `shows_next_episode()`

Examples

```
seasons_episodes("breaking-bad", 1)

# Including all episode data:
seasons_episodes("breaking-bad", 1, extended = "full")
```

---

seasons_season *Get a season of a show*

Description

Similar to seasons_summary, but this function returns only metadata for a season.

Usage

```
seasons_season(id, seasons = 1L, extended = "min")
```

Arguments

`id`	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
`seasons`	`integer(1)` `[1L]`: The season(s) to get. Use `0` for specials.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc.

- `"images"`: Minimal info plus image URLs (returned as a list-column).
- `"full,images"`: Complete info plus images.
- `"metadata"`: Collection endpoints only; adds video/audio metadata.

Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Note

If you want to quickly gather episode data of all available seasons, see seasons_summary and use the `episodes = TRUE` parameter.

Source

`seasons_season()` wraps endpoint /shows/:id/seasons/:season/info.

See Also

Other season data: `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `seasons_episodes()`, `seasons_summary()`

Examples

```
seasons_season("breaking-bad", 1)

# Including all episode data:
seasons_season("breaking-bad", 1, extended = "full")
```

`seasons_summary` *Get a show's seasons*

Description

Get details for a show's seasons, e.g. how many seasons there are and how many epsiodes each season has. With `episodes == TRUE` and `extended == "full"`, this function is also suitable to retrieve all episode data for all seasons of a show with just a single API call.

Usage

```
seasons_summary(
  id,
  episodes = FALSE,
  drop_specials = TRUE,
  drop_unaired = TRUE,
  extended = "min"
)
```

Arguments

id	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
episodes	`logical(1)` `[FALSE]`: If `TRUE`, all episodes for each season are appended as a list-column, with the amount of variables depending on `extended`.
drop_specials	`logical(1)` `[TRUE]`: Special episodes (season 0) are dropped
drop_unaired	`logical(1)` `[TRUE]`: Seasons without aired episodes are dropped. Only works if `extended` is `"full"`.
extended	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`seasons_summary()` wraps endpoint /shows/:id/seasons/:season.

See Also

Other season data: `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `seasons_episodes()`, `seasons_season()`

Other episode data: episodes_summary(), media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, seasons_episodes(), shows_next_episode()

Other summary methods: comments_comment(), episodes_summary(), movies_summary(), people_summary(), shows_summary(), user_profile()

Examples

```
# Get just the season numbers and their IDs
seasons_summary("breaking-bad", extended = "min")
## Not run:
# Get season numbers, ratings, votes, titles and other metadata as well as
# a list-column containing all episode data
seasons_summary("utopia", extended = "full", episodes = TRUE)

## End(Not run)
```

---

shows_next_episode *Get a shows next or latest episode*

Description

Get a shows next or latest episode

Usage

```
shows_next_episode(id, extended = "min")

shows_last_episode(id, extended = "min")
```

Arguments

id	character(1): The ID of the item requested. Preferably the trakt ID (e.g. 1429). Other options are the trakt.tv slug (e.g. "the-wire") or imdb ID (e.g. "tt0306414"). Can also be of length greater than 1, in which case the function is called on all id values separately and the result is combined. See vignette("tRakt") for more details.
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`shows_next_episode()` wraps endpoint /shows/:id/next_episode.

`shows_last_episode()` wraps endpoint /shows/:id/last_episode.

See Also

Other show data: `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `people_media()`, `played_media`, `shows_related()`, `shows_summary()`, `updated_media`

Other episode data: `episodes_summary()`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `seasons_episodes()`, `seasons_summary()`

Examples

```
shows_next_episode("one-piece")
shows_last_episode("one-piece", extended = "full")
```

---

shows_related *Get similiar(ish) shows*

Description

Get similiar(ish) shows

Usage

```
shows_related(id, limit = 10L, extended = "min")
```

Arguments

`id`	`character(1)`: The ID of the item requested. Preferably the `trakt` ID (e.g. `1429`). Other options are the trakt.tv `slug` (e.g. `"the-wire"`) or `imdb` ID (e.g. `"tt0306414"`). Can also be of length greater than 1, in which case the function is called on all `id` values separately and the result is combined. See `vignette("tRakt")` for more details.
`limit`	`integer(1)` `[10L]`: Number of items to return. Must be greater than `0` and will be coerced via `as.integer()`.
`extended`	`character`: Level of detail for the API response.

- `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param.
- `"full"`: Complete info including overview, ratings, runtime, etc.
- `"images"`: Minimal info plus image URLs (returned as a list-column).
- `"full,images"`: Complete info plus images.
- `"metadata"`: Collection endpoints only; adds video/audio metadata.

Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`shows_related()` wraps endpoint /shows/:id/related.

See Also

Other show data: `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `people_media()`, `played_media`, `shows_next_episode()`, `shows_summary()`, `updated_media`

Examples

```
shows_related("breaking-bad", limit = 5)
```

`shows_summary` *Get a single show*

Description

Get a single show

Usage

```
shows_summary(id, extended = "min")
```

Arguments

id	character(1): The ID of the item requested. Preferably the trakt ID (e.g. 1429). Other options are the trakt.tv slug (e.g. "the-wire") or imdb ID (e.g. "tt0306414"). Can also be of length greater than 1, in which case the function is called on all id values separately and the result is combined. See vignette("tRakt") for more details.
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Source

shows_summary() wraps endpoint /shows/:id.

See Also

Other show data: collected_media, media_aliases, media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, people_media(), played_media, shows_next_episode(), shows_related(), updated_media

Other summary methods: comments_comment(), episodes_summary(), movies_summary(), people_summary(), seasons_summary(), user_profile()

Examples

```
# Minimal info by default
shows_summary("breaking-bad")
## Not run:
# More information
shows_summary("breaking-bad", extended = "full")

## End(Not run)
```

trakt_api_available *Check whether the trakt.tv API is reachable*

Description

A lightweight connectivity check that performs a single, small request to a stable endpoint with a short timeout. It is primarily used to guard runnable documentation examples (via roxygen2's `@examplesIf`) so they are executed when the API is available but quietly skipped when it is not — avoiding spurious failures from transient API outages or a missing internet connection.

Usage

```
trakt_api_available()
```

Details

This function never throws: it returns `FALSE` on any error (offline, timeout, non-success status, ...) and does not retry.

Value

`logical(1)`: `TRUE` if the API responds successfully, else `FALSE`.

See Also

Other API-basics: `trakt_get()`

Examples

```
# Returns TRUE or FALSE, never errors:
trakt_api_available()
```

trakt_credentials *trakt.tv credentials*

Description

Authentication has changed with the migration to httr2, so this function is just a placeholder that shows information about available credentials.

Usage

```
trakt_credentials()
```

`trakt_datasets` *Cached filter datasets*

Description

These datasets are used internally to check the optional filter parameters for certain functions (see search_query or the dynamic lists like shows_popular). They are unlikely to change often and are therefore included as package datasets.

Usage

```
trakt_genres

trakt_languages

trakt_networks

trakt_countries

trakt_certifications
```

Format

Every dataset is a tibble(). The following list includes the dataset topic with a link to the API documentation, a short description and a list of variables with example values:

- Genres: Genres for shows and movies (with their two-letter codes) trakt.tv knows.
  - 3 Variables: `name` ("Action"), `slug` ("action"), `type` ("movies")
- Languages: Languages (and two-letter codes) trakt.tv knows.
  - 3 Variables: `name` ("Arabic"), `code` ("ar"), `type` ("movies")
- Networks: TV networks trakt.tv knows.
  - 2 Variables: `name` ("TBS"), `name_clean` ("tbs") (lower-case, no trailing whitespaces)
- Countries: Country names (and two-letter codes).
  - 3 Variables: `name` ("Belarus"), `code` ("by"), `type` ("movies")
- Certifications: TV and movie certifications (e.g. "PG-13" and the likes).
  - 5 Variables: `country` ("us" only), `name` ("TV-PG"), `slug` ("tv-pg"), `description` ("Parental Guidance Suggested"), `type` ("shows")

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 64 rows and 3 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 286 rows and 3 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 4682 rows and 5 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 384 rows and 3 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 12 rows and 5 columns.

Details

The datasets are prefixed with `trakt_` purely to avoid confusion or masking for filter arguments of the same name.

Note

Currently only US certifications are available.

Examples

```
head(trakt_genres)
head(trakt_languages)
head(trakt_networks)
head(trakt_countries)
trakt_certifications
```

---

trakt_get *Make an API call and receive parsed output*

Description

The most basic form of API interaction: Querying a specific URL and getting its parsed result. If the response is empty, the function returns an empty tibble(), and if there are date-time variables present in the response, they are converted to `POSIXct` via `lubridate::ymd_hms()` or to `Date` via `lubridate::as_date()` if the variable only contains date information.

Usage

```
trakt_get(url)
```

Arguments

`url`	`character(1)`: The API endpoint. Either a full URL like `"https://api.trakt.tv/shows/breaking-b` or just the endpoint like `shows/breaking-bad`.

Details

See the official API reference for a detailed overview of available methods. Most methods of potential interest for data collection have dedicated functions in this package.

Value

The parsed content of the API response. An empty tibble() if the response is an empty `JSON` array.

See Also

Other API-basics: `trakt_api_available()`

Examples

```
# A simple request to a direct URL
trakt_get("https://api.trakt.tv/shows/breaking-bad")

# Optionally be lazy about URL specification by dropping the hostname:
trakt_get("shows/game-of-thrones")
```

`tRakt_sitrep` *tRakt sitrep*

Description

Get general information about set options and API credentaials.

Usage

```
tRakt_sitrep()
```

Examples

```
tRakt_sitrep()
```

`trending_media` *Trending media*

Description

These functions return the trending movies/shows on trakt.tv.

Usage

```
movies_trending(
  limit = 10,
  extended = "min",
  filters = NULL,
  query = NULL,
  years = NULL,
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL
)
```

```
shows_trending(
  limit = 10,
  extended = "min",
  filters = NULL,
  query = NULL,
  years = NULL,
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL,
  networks = NULL,
  status = NULL
)
```

Arguments

limit	integer(1) [10L]: Number of items to return. Must be greater than 0 and will be coerced via as.integer().
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).
filters	A trakt_filters object created with filters_movies(), filters_shows(), or filters_episodes() that refines which items are returned. See filters for the full set of supported filters. Supplying filters as individual arguments (genres, years, networks, ...) is soft-deprecated in favour of this argument; if both are given, filters takes precedence.
query	character(1): Search string for titles and descriptions. For search_query() other fields are searched depending on the type of media. See the API docs for a full reference.
years	character | integer: 4-digit year (2010) **or** range, e.g. "2010-2020". Can also be an integer vector of length two which will be coerced appropriately, e.g. c(2010, 2020).
genres	character(n): Genre slug(s). See trakt_genres for a table of genres. Multiple values are allowed and will be concatenated.
languages	character(n): Two-letter language code(s). Also see trakt_languages for available languages (code and name).
countries	character(n): Two-letter country code(s). See trakt_countries.

runtimes	character | integer: Integer range in minutes, e.g. 30-90. Can also be an integer vector of length two which will be coerced appropriately.
ratings	character | integer: Integer range between 0 and 100. Can also be an integer vector of length two which will be coerced appropriately. Note that user-supplied ratings are in the range of 1 to 10, yet the ratings on the site itself are scaled to the range of 1 to 100.
certifications	character(n): Certification(s) like pg-13. Multiple values are allowed. Use trakt_certifications for reference. Note that there are different certifications for shows and movies.
networks	character(n): (Shows only) Network name like HBO. See trakt_networks for a list of known networks.
status	character(n): (Shows only) The status of the shows. One of "returning series", "in production", "planned", "canceled", or "ended".

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

The Dynamic Lists on trakt.tv

These functions access the automatically updated lists provided by trakt.tv. Each function comes in two flavors: Shows or movies. The following descriptions are adapted directly from the API reference.

- Popular: Popularity is calculated using the rating percentage and the number of ratings.
- Trending: Returns all movies/shows being watched right now. Movies/shows with the most users are returned first.
- Played: Returns the most played (a single user can watch multiple times) movies/shows in the specified time period.
- Watched: Returns the most watched (unique users) movies/shows in the specified time period.
- Collected: Returns the most collected (unique users) movies/shows in the specified time period.
- Anticipated: Returns the most anticipated movies/shows based on the number of lists a movie/show appears on. The functions for **Played**, **Watched**, **Collected** and **Played** each return the same additional variables besides the media information: watcher_count, play_count, collected_count, collector_count.

Source

movies_trending() wraps endpoint /movies/trending.

shows_trending() wraps endpoint /shows/trending.

See Also

Other movie data: anticipated_media, collected_media, media_aliases, media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching, movies_boxoffice(), movies_related(), movies_releases(), movies_summary(), people_media(), played_media, popular_media, updated_media, watched_media

Other dynamic lists: anticipated_media, collected_media, filters, lists_popular(), played_media, popular_media, updated_media, watched_media

Other shows data: anticipated_media, popular_media, watched_media

---

`updated_media` *Recently updated media*

Description

Return movies or shows that were updated on trakt.tv since `start_date`. Handy for keeping a local cache in sync: store the most recent `updated_at` you have seen and poll for anything newer.

Usage

```
movies_updates(limit = 10, extended = "min", start_date = Sys.Date() - 1)

shows_updates(limit = 10, extended = "min", start_date = Sys.Date() - 1)
```

Arguments

`limit`	`integer(1) [10L]`: Number of items to return. Must be greater than `0` and will be coerced via `as.integer()`.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).
`start_date`	`Date | character(1)`: Return items updated since this date. Defaults to yesterday. The trakt.tv API only accepts dates up to **30 days** in the past; older dates return no results (a warning is emitted).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

The Dynamic Lists on trakt.tv

These functions access the automatically updated lists provided by trakt.tv. Each function comes in two flavors: Shows or movies. The following descriptions are adapted directly from the API reference.

- Popular: Popularity is calculated using the rating percentage and the number of ratings.
- Trending: Returns all movies/shows being watched right now. Movies/shows with the most users are returned first.
- Played: Returns the most played (a single user can watch multiple times) movies/shows in the specified time `period`.
- Watched: Returns the most watched (unique users) movies/shows in the specified time `period`.
- Collected: Returns the most collected (unique users) movies/shows in the specified time `period`.
- Anticipated: Returns the most anticipated movies/shows based on the number of lists a movie/show appears on. The functions for **Played**, **Watched**, **Collected** and **Played** each return the same additional variables besides the media information: `watcher_count`, `play_count`, `collected_count`, `collector_count`.

Note

Unlike the other dynamic lists, the updates endpoints do not support the `filters` argument.

Source

`movies_updates()` wraps endpoint /movies/updates/:start_date.

`shows_updates()` wraps endpoint /shows/updates/:start_date.

See Also

Other movie data: `anticipated_media`, `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `movies_boxoffice()`, `movies_related()`, `movies_releases()`, `movies_summary()`, `people_media()`, `played_media`, `popular_media`, `trending_media`, `watched_media`

Other dynamic lists: `anticipated_media`, `collected_media`, `filters`, `lists_popular()`, `played_media`, `popular_media`, `trending_media`, `watched_media`

Other show data: `collected_media`, `media_aliases`, `media_comments`, `media_lists`, `media_people`, `media_ratings()`, `media_stats()`, `media_translations`, `media_watching`, `people_media()`, `played_media`, `shows_next_episode()`, `shows_related()`, `shows_summary()`

Examples

```
movies_updates()
shows_updates(start_date = Sys.Date() - 7)
```

`user_collection` *Get a user's collected shows or movies*

Description

Get a user's collected shows or movies

Usage

```
user_collection(
  user = "me",
  type = c("shows", "movies"),
  unnest_episodes = FALSE,
  extended = "min"
)
```

Arguments

`user`	`character(1)`: Target username (or `slug`). Defaults to `"me"`, the OAuth user. Can also be of length greater than 1, in which case the function is called on all `user` values separately and the result is combined.
`type`	`character(1)`: Either `"shows"` or `"movies"`. For season/episode-specific functions, values `seasons` or `episodes` are also allowed.
`unnest_episodes`	`logical(1)` `[FALSE]`: Unnests episode data using `tidyr::unnest()` and returns one row per episode rather than one row per show.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Details

This function wraps the API method `/users/:user_id/collection/:type`.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Note

The `extended = "metadata"` API parameter can be used to add media information like `media_type`, `resolution`, `audio`, `audio_channels` and `3D` to the output. Combine with `"full"` as `extended = "full,metadata"`.

Source

`user_collection()` wraps endpoint /users/:id/collection/:type.

See Also

Other user data: `user_comments()`, `user_history()`, `user_likes()`, `user_network()`, `user_profile()`, `user_ratings()`, `user_stats()`, `user_watched()`, `user_watchlist()`

Examples

```
## Not run:
user_collection(user = "sean", type = "movies")
user_collection(user = "sean", type = "shows")

## End(Not run)
```

---

user_comments *Get a user's comments*

Description

Get a user's comments

Usage

```
user_comments(
  user = "me",
  comment_type = c("all", "reviews", "shouts"),
  type = c("all", "movies", "shows", "seasons", "episodes", "lists"),
  include_replies = FALSE
)
```

Arguments

`user`	`character(1)`: Target username (or `slug`). Defaults to `"me"`, the OAuth user. Can also be of length greater than 1, in which case the function is called on all `user` values separately and the result is combined.
`comment_type`	`character(1)` `["all"]`: The type of comment, one of "all", "reviews" or "shouts".
`type`	`character(1)` `["all"]`: The type of media to filter by, one of "all", "movies", "shows", "seasons", "episodes" or "lists".
`include_replies`	`logical(1)` `[FALSE]`: Whether to include replies.

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`user_comments()` wraps endpoint /users/:id/comments/:comment_type/:type?include_replies=.

See Also

Other user data: `user_collection()`, `user_history()`, `user_likes()`, `user_network()`, `user_profile()`, `user_ratings()`, `user_stats()`, `user_watched()`, `user_watchlist()`

Other comment methods: `comments_comment()`, `comments_trending()`, `comments_updates()`, `media_comments`, `user_list_comments()`

Examples

```
## Not run:
user_comments("jemus42")

## End(Not run)
```

`user_history` *Get a user's watch history*

Description

Retrieve a the last `limit` items a user has watched, including the method by which it was logged (e.g. *scrobble* or *checkin*).

Usage

```
user_history(
  user = "me",
  type = c("shows", "movies", "seasons", "episodes"),
  item_id = NULL,
  limit = 10L,
  start_at = NULL,
  end_at = NULL,
  extended = "min"
)
```

Arguments

user	character(1): Target username (or slug). Defaults to "me", the OAuth user. Can also be of length greater than 1, in which case the function is called on all user values separately and the result is combined.
type	character(1): Either "shows" or "movies". For season/episode-specific functions, values seasons or episodes are also allowed.
item_id	character(1): The ID of the item you're looking for.
limit	integer(1) [10L]: Number of items to return. Must be greater than 0 and will be coerced via as.integer().
start_at, end_at	character(1): A time-window to filter by. Must be coercible to a datetime object of class POSIXct. See ISOdate for further information.
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).

Details

This function wraps the API method /users/:id/history/:type.

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Note

For type = "shows", the original output contains a nested object with show and episode data, which are unnested by this function. Due to duplicate variable names, all episode-related variables are prefixed with episode_. This results in the episode number having the name episode_episode, which is quite silly. Sorry.

Source

user_history() wraps endpoint /users/:id/history/:type/:item_id?start_at=,end_at=.

See Also

Other user data: user_collection(), user_comments(), user_likes(), user_network(), user_profile(), user_ratings(), user_stats(), user_watched(), user_watchlist()

Examples

```
## Not run:
# Shows user "jemus42" watched around christmas 2016
user_history(
  user = "jemus42", type = "shows", limit = 5,
  start_at = "2015-12-24", end_at = "2015-12-28"
)

## End(Not run)
```

`user_likes` *Get items (comments, lists) a user likes*

Description

Get items (comments, lists) a user likes

Usage

```
user_likes(type = c("comments", "lists"))
```

Arguments

`type`	`character(1)` `["comments"]`: One of "comments", "lists".

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`user_likes()` wraps endpoint /users/:id/likes/:type (Authentication required).

See Also

Other user data: `user_collection()`, `user_comments()`, `user_history()`, `user_network()`, `user_profile()`, `user_ratings()`, `user_stats()`, `user_watched()`, `user_watchlist()`

Examples

```
# Get liked lists (only if there's a client secret set)
user_likes("lists")
```

`user_list` *Get a single list*

Description

Retrieve a single list a user has created together with information about the user. Use `extended = "full"` to retrieve all user profile data, similiar to user_profile. The returned variables `trakt` (list ID) or `slug` (list slug) can be used to retrieve the list's items via user_list_items.

Usage

```
user_list(user = "me", list_id, extended = "min")
```

Arguments

`user`	`character(1)`: Target username (or `slug`). Defaults to `"me"`, the OAuth user. Can also be of length greater than 1, in which case the function is called on all `user` values separately and the result is combined.
`list_id`	The list identifier, either `trakt` ID or `slug` of the list. Can be optained via the website (URL `slug`) or e.g. user_lists.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Note

In the embedded user data, `name` is renamed to `user_name` due to duplication with e.g. list names, and `slug` is renamed to `user_slug` for the same reason.

See Also

user_list for only a single list.

user_list_items For the actual content of a list.

Other list methods: `lists_popular()`, `media_lists`, `user_list_comments()`, `user_list_items()`, `user_lists()`

Examples

```
## Not run:
user_list("jemus42", list_id = 2121308)

## End(Not run)
```

`user_list_comments` *Get comments on a user-created list*

Description

Get comments on a user-created list

Usage

```
user_list_comments(
  user = "me",
  list_id,
  sort = c("newest", "oldest", "likes", "replies"),
  extended = "min"
)
```

Arguments

`user`	`character(1)`: Target username (or `slug`). Defaults to `"me"`, the OAuth user. Can also be of length greater than 1, in which case the function is called on all `user` values separately and the result is combined.
`list_id`	The list identifier, either `trakt` ID or `slug` of the list. Can be optained via the website (URL `slug`) or e.g. user_lists.
`sort`	`character(1)` `["newest"]`: Comment sort order, one of "newest", "oldest", "likes" or "replies".
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`user_list_comments()` wraps endpoint /users/:id/lists/:list_id/comments/:sort.

See Also

Other comment methods: `comments_comment()`, `comments_trending()`, `comments_updates()`, `media_comments`, `user_comments()`

Other list methods: `lists_popular()`, `media_lists`, `user_list()`, `user_list_items()`, `user_lists()`

Examples

```
## Not run:
user_list_comments("donxy", "1248149")

## End(Not run)
```

---

user_list_items *Get a user's list's items*

Description

Get a user's list's items

Usage

```
user_list_items(user = "me", list_id, type = NULL, extended = "min")
```

Arguments

`user`	`character(1)`: Target username (or `slug`). Defaults to `"me"`, the OAuth user. Can also be of length greater than 1, in which case the function is called on all `user` values separately and the result is combined.
`list_id`	The list identifier, either `trakt` ID or `slug` of the list. Can be optained via the website (URL `slug`) or e.g. user_lists.
`type`	`character(1) [NULL]`: If not `NULL`, only items of that media type are returned. Possible values are `"movie"`, `"show"`, `"season"`, `"episode"`, `"person"`.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`user_list_items()` wraps endpoint /users/:id/lists/:list_id/items/:type.

See Also

Other list methods: `lists_popular()`, `media_lists`, `user_list()`, `user_list_comments()`, `user_lists()`

Examples

```
## Not run:
# A large list with various media types
# All items
user_list_items("sp1ti", list_id = "5615781", extended = "min")

# Movies only
user_list_items("sp1ti", list_id = "5615781", extended = "min", type = "movie")

# Shows...
user_list_items("sp1ti", list_id = "5615781", extended = "min", type = "shows")

# Only seasons
user_list_items("sp1ti", list_id = "5615781", extended = "min", type = "season")

# Only episodes
user_list_items("sp1ti", list_id = "5615781", extended = "min", type = "episodes")

## End(Not run)
```

---

user_lists *Get a user's lists*

Description

Retrieve all lists a user has created together with information about the user. Use `extended = "full"` to retrieve all user profile data, similiar to user_profile. The returned variables `trakt` (list ID) or `slug` (list slug) can be used to retrieve the list's items via user_list_items.

Usage

```
user_lists(user = "me", extended = "min")
```

Arguments

user	character(1): Target username (or slug). Defaults to "me", the OAuth user. Can also be of length greater than 1, in which case the function is called on all user values separately and the result is combined.
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Note

In the embedded user data, name is renamed to user_name due to duplication with e.g. list names, and slug is renamed to user_slug for the same reason.

Source

user_lists() wraps endpoint /users/:id/lists.

See Also

user_lists for all lists a user has.

user_list_items For the actual content of a list.

Other list methods: lists_popular(), media_lists, user_list(), user_list_comments(), user_list_items()

Examples

```
## Not run:
user_lists("jemus42")

## End(Not run)
```

`user_network` *Get a user's social connections*

Description

Get followers, followings or friends (the two-way relationship).

Usage

```
user_followers(user = getOption("trakt_user"), extended = "min")

user_following(user = getOption("trakt_user"), extended = "min")

user_friends(user = getOption("trakt_user"), extended = "min")
```

Arguments

`user`	`character(1)`: Target username (or `slug`). Defaults to `"me"`, the OAuth user. Can also be of length greater than 1, in which case the function is called on all `user` values separately and the result is combined.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Note

If the specified user is private, you need to be able to make an authenticated request and be friends with the user.

Source

`user_followers()` wraps endpoint /users/:id/followers.

`user_following()` wraps endpoint /users/:id/following.

`user_friends()` wraps endpoint /users/:id/friends.

See Also

Other user data: `user_collection()`, `user_comments()`, `user_history()`, `user_likes()`, `user_profile()`, `user_ratings()`, `user_stats()`, `user_watched()`, `user_watchlist()`

Examples

```
## Not run:
user_followers(user = "sean")

## End(Not run)
## Not run:
user_following(user = "sean")

## End(Not run)
## Not run:
user_friends(user = "sean")

## End(Not run)
```

`user_profile` *Get a user's profile*

Description

Get a user's profile

Usage

```
user_profile(user = "me", extended = "min")
```

Arguments

`user`	`character(1)`: Target username (or `slug`). Defaults to `"me"`, the OAuth user. Can also be of length greater than 1, in which case the function is called on all `user` values separately and the result is combined.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Note

If the specified user is private, you need to be able to make an authenticated request and be friends with the user.

Source

`user_profile()` wraps endpoint /users/:id.

See Also

Other user data: `user_collection()`, `user_comments()`, `user_history()`, `user_likes()`, `user_network()`, `user_ratings()`, `user_stats()`, `user_watched()`, `user_watchlist()`

Other summary methods: `comments_comment()`, `episodes_summary()`, `movies_summary()`, `people_summary()`, `seasons_summary()`, `shows_summary()`

Examples

```
## Not run:
user_profile("sean")

## End(Not run)
```

---

user_ratings *Get a user's ratings*

Description

Retrieve a user's media ratings

Usage

```
user_ratings(
  user = "me",
  type = c("movies", "seasons", "shows", "episodes"),
  rating = NULL,
  extended = "min",
  limit = NULL
)
```

Arguments

user	character(1): Target username (or slug). Defaults to "me", the OAuth user. Can also be of length greater than 1, in which case the function is called on all user values separately and the result is combined.
type	character(1): Either "shows" or "movies". For season/episode-specific functions, values seasons or episodes are also allowed.
rating	integer(1) [NULL]: Optional rating between 1 and 10 to filter by.
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).
limit	integer(1) [NULL]: Number of items to return. If NULL (default), all items are returned.

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Source

user_ratings() wraps endpoint /users/:id/ratings/:type/:rating.

See Also

Other user data: user_collection(), user_comments(), user_history(), user_likes(), user_network(), user_profile(), user_stats(), user_watched(), user_watchlist()

Examples

```
## Not run:
user_ratings(user = "jemus42", "shows")
user_ratings(user = "sean", type = "movies")

## End(Not run)
```

`user_stats` *Returns stats about the movies, shows, and episodes a user has watched, collected, and rated.*

Description

Data about a user's interactions with movies, shows, seasons, episodes, as well as their social network (friends, followings, followers) and a frequency table of the user's media ratings so far.

Usage

```
user_stats(user = "me")
```

Arguments

`user`	`character(1)`: Target username (or `slug`). Defaults to `"me"`, the OAuth user. Can also be of length greater than 1, in which case the function is called on all `user` values separately and the result is combined.

Value

A `list` of tibbles containing the following elements:

- "movies"
- "shows"
- "seasons"
- "episodes"
- "network"
- "ratings"

Source

`user_stats()` wraps endpoint /users/:id/stats.

See Also

Other user data: `user_collection()`, `user_comments()`, `user_history()`, `user_likes()`, `user_network()`, `user_profile()`, `user_ratings()`, `user_watched()`, `user_watchlist()`

Examples

```
## Not run:
user_stats(user = "sean")

## End(Not run)
```

`user_watched` *Get a user's watched shows or movies*

Description

For private users, an authenticated request is required.

Usage

```
user_watched(
  user = "me",
  type = c("shows", "movies"),
  noseasons = TRUE,
  extended = "min"
)
```

Arguments

`user`	`character(1)`: Target username (or `slug`). Defaults to `"me"`, the OAuth user. Can also be of length greater than 1, in which case the function is called on all `user` values separately and the result is combined.
`type`	`character(1)`: Either `"shows"` or `"movies"`. For season/episode-specific functions, values `seasons` or `episodes` are also allowed.
`noseasons`	`logical(1)` `[TRUE]`: Only for `type = "show"`: Exclude detailed season data from output. This is advisable if you do not need per-episode data and want to be nice to the API.
`extended`	`character`: Level of detail for the API response. • `"min"` (default): Minimal info (title, year, IDs). Omits the `extended` query param. • `"full"`: Complete info including overview, ratings, runtime, etc. • `"images"`: Minimal info plus image URLs (returned as a list-column). • `"full,images"`: Complete info plus images. • `"metadata"`: Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. `"full,images"`) or a character vector (e.g. `c("full", "images")`).

Value

A tibble(). If the function has a `limit` parameter (defaulting to `10`), this will be the (maximum) number of rows of the `tibble`. If there are no results (or the API is unreachable), an empty `tibble()` is returned.

Source

`user_watched()` wraps endpoint /users/:id/watched/:type.

See Also

Other user data: user_collection(), user_comments(), user_history(), user_likes(), user_network(), user_profile(), user_ratings(), user_stats(), user_watchlist()

Examples

```
## Not run:
# Use noseasons = TRUE to avoid receiving detailed season/episode data
user_watched(user = "sean", noseasons = TRUE)

## End(Not run)
```

---

user_watchlist *Get a user's watchlist*

Description

Get a user's watchlist

Usage

```
user_watchlist(user = "me", type = c("movies", "shows"), extended = "min")
```

Arguments

user	character(1): Target username (or slug). Defaults to "me", the OAuth user. Can also be of length greater than 1, in which case the function is called on all user values separately and the result is combined.
type	character(1): Either "shows" or "movies". For season/episode-specific functions, values seasons or episodes are also allowed.
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

Source

user_watchlist() wraps endpoint /users/:id/watchlist/:type/:sort.

See Also

Other user data: user_collection(), user_comments(), user_history(), user_likes(), user_network(), user_profile(), user_ratings(), user_stats(), user_watched()

Examples

```
## Not run:
# Defaults to movie watchlist and minimal info
user_watchlist(user = "sean")

## End(Not run)
```

---

watched_media *Most watched media*

Description

These functions return the most watched movies/shows on trakt.tv.

Usage

```
movies_watched(
  limit = 10,
  extended = "min",
  period = c("weekly", "monthly", "yearly", "all"),
  filters = NULL,
  query = NULL,
  years = NULL,
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL
)

shows_watched(
  limit = 10,
  extended = "min",
  period = c("weekly", "monthly", "yearly", "all"),
  filters = NULL,
  query = NULL,
  years = NULL,
```

```
  genres = NULL,
  languages = NULL,
  countries = NULL,
  runtimes = NULL,
  ratings = NULL,
  certifications = NULL,
  networks = NULL,
  status = NULL
)
```

Arguments

limit	integer(1) [10L]: Number of items to return. Must be greater than 0 and will be coerced via as.integer().
extended	character: Level of detail for the API response. • "min" (default): Minimal info (title, year, IDs). Omits the extended query param. • "full": Complete info including overview, ratings, runtime, etc. • "images": Minimal info plus image URLs (returned as a list-column). • "full,images": Complete info plus images. • "metadata": Collection endpoints only; adds video/audio metadata. Multiple values can be combined as a comma-separated string (e.g. "full,images") or a character vector (e.g. c("full", "images")).
period	character(1) ["weekly"]: Which period to filter by. Possible values are "weekly", "monthly", "yearly", "all".
filters	A trakt_filters object created with filters_movies(), filters_shows(), or filters_episodes() that refines which items are returned. See filters for the full set of supported filters. Supplying filters as individual arguments (genres, years, networks, ...) is soft-deprecated in favour of this argument; if both are given, filters takes precedence.
query	character(1): Search string for titles and descriptions. For search_query() other fields are searched depending on the type of media. See the API docs for a full reference.
years	character | integer: 4-digit year (2010) **or** range, e.g. "2010-2020". Can also be an integer vector of length two which will be coerced appropriately, e.g. c(2010, 2020).
genres	character(n): Genre slug(s). See trakt_genres for a table of genres. Multiple values are allowed and will be concatenated.
languages	character(n): Two-letter language code(s). Also see trakt_languages for available languages (code and name).
countries	character(n): Two-letter country code(s). See trakt_countries.
runtimes	character | integer: Integer range in minutes, e.g. 30-90. Can also be an integer vector of length two which will be coerced appropriately.

ratings	character | integer: Integer range between 0 and 100. Can also be an integer vector of length two which will be coerced appropriately. Note that user-supplied ratings are in the range of 1 to 10, yet the ratings on the site itself are scaled to the range of 1 to 100.
certifications	character(n): Certification(s) like pg-13. Multiple values are allowed. Use trakt_certifications for reference. Note that there are different certifications for shows and movies.
networks	character(n): (Shows only) Network name like HBO. See trakt_networks for a list of known networks.
status	character(n): (Shows only) The status of the shows. One of "returning series", "in production", "planned", "canceled", or "ended".

Value

A tibble(). If the function has a limit parameter (defaulting to 10), this will be the (maximum) number of rows of the tibble. If there are no results (or the API is unreachable), an empty tibble() is returned.

The Dynamic Lists on trakt.tv

These functions access the automatically updated lists provided by trakt.tv. Each function comes in two flavors: Shows or movies. The following descriptions are adapted directly from the API reference.

- Popular: Popularity is calculated using the rating percentage and the number of ratings.
- Trending: Returns all movies/shows being watched right now. Movies/shows with the most users are returned first.
- Played: Returns the most played (a single user can watch multiple times) movies/shows in the specified time period.
- Watched: Returns the most watched (unique users) movies/shows in the specified time period.
- Collected: Returns the most collected (unique users) movies/shows in the specified time period.
- Anticipated: Returns the most anticipated movies/shows based on the number of lists a movie/show appears on. The functions for **Played**, **Watched**, **Collected** and **Played** each return the same additional variables besides the media information: watcher_count, play_count, collected_count, collector_count.

Source

movies_watched() wraps endpoint /movies/watched/:period.

shows_watched() wraps endpoint /shows/watched/:period.

See Also

Other movie data: anticipated_media, collected_media, media_aliases, media_comments, media_lists, media_people, media_ratings(), media_stats(), media_translations, media_watching,

movies_boxoffice(), movies_related(), movies_releases(), movies_summary(), people_media(), played_media, popular_media, trending_media, updated_media

Other dynamic lists: anticipated_media, collected_media, filters, lists_popular(), played_media, popular_media, trending_media, updated_media

Other shows data: anticipated_media, popular_media, trending_media

Index

∗ **API-basics**
trakt_api_available, 59
trakt_get, 61
∗ **Episode datasets**
futurama, 18
gameofthrones, 19
∗ **comment methods**
comments_comment, 9
comments_trending, 11
comments_updates, 13
media_comments, 22
user_comments, 68
user_list_comments, 73
∗ **datasets**
futurama, 18
gameofthrones, 19
trakt_datasets, 60
∗ **dynamic lists**
anticipated_media, 3
collected_media, 7
filters, 15
lists_popular, 20
played_media, 43
popular_media, 45
trending_media, 62
updated_media, 65
watched_media, 84
∗ **episode data**
episodes_summary, 14
media_comments, 22
media_lists, 24
media_people, 27
media_ratings, 29
media_stats, 30
media_translations, 32
media_watching, 33
seasons_episodes, 51
seasons_summary, 53
shows_next_episode, 55
∗ **list methods**
lists_popular, 20
media_lists, 24
user_list, 72
user_list_comments, 73
user_list_items, 74
user_lists, 75
∗ **movie data**
anticipated_media, 3
collected_media, 7
media_aliases, 21
media_comments, 22
media_lists, 24
media_people, 27
media_ratings, 29
media_stats, 30
media_translations, 32
media_watching, 33
movies_boxoffice, 35
movies_related, 36
movies_releases, 37
movies_summary, 38
people_media, 40
played_media, 43
popular_media, 45
trending_media, 62
updated_media, 65
watched_media, 84
∗ **people data**
media_lists, 24
media_people, 27
people_media, 40
people_summary, 41
∗ **search functions**
search_query, 48
∗ **season data**
media_comments, 22
media_lists, 24
media_people, 27

media_ratings, 29
media_stats, 30
seasons_episodes, 51
seasons_season, 52
seasons_summary, 53
∗ **show data**
collected_media, 7
media_aliases, 21
media_comments, 22
media_lists, 24
media_people, 27
media_ratings, 29
media_stats, 30
media_translations, 32
media_watching, 33
people_media, 40
played_media, 43
shows_next_episode, 55
shows_related, 56
shows_summary, 57
updated_media, 65
∗ **shows data**
anticipated_media, 3
popular_media, 45
trending_media, 62
watched_media, 84
∗ **summary methods**
comments_comment, 9
episodes_summary, 14
movies_summary, 38
people_summary, 41
seasons_summary, 53
shows_summary, 57
user_profile, 78
∗ **user data**
user_collection, 67
user_comments, 68
user_history, 69
user_likes, 71
user_network, 77
user_profile, 78
user_ratings, 79
user_stats, 81
user_watched, 82
user_watchlist, 83
∗ **utility functions**
build_trakt_url, 6
pad_episode, 39

Anticipated, *5*, *9*, *45*, *48*, *64*, *66*, *86*
anticipated_media, 3, *9*, *17*, *20, 21*, *23*, *26*, *28*, *30, 31*, *33, 34*, *36–39*, *41*, *45*, *48*, *65, 66*, *86, 87*
authenticated request, *77*, *79*, *82*

build_trakt_url, 6
build_trakt_url(), *40*

certification slugs, *16*
Collected, *5*, *9*, *45*, *48*, *64*, *66*, *86*
collected_media, *5*, 7, *17*, *20, 21*, *23, 24*, *26*, *28*, *30, 31*, *33–39*, *41*, *45*, *48*, *56–58*, *65, 66*, *86, 87*
comments_comment, 9
comments_comment(), *12, 13*, *15*, *23*, *39*, *42*, *55*, *58*, *69*, *74*, *79*
comments_item *(*comments_comment*)*, 9
comments_likes *(*comments_comment*)*, 9
comments_recent *(*comments_trending*)*, 11
comments_replies *(*comments_comment*)*, 9
comments_trending, 11
comments_trending(), *11*, *13*, *23*, *69*, *74*
comments_updates, 13
comments_updates(), *11, 12*, *23*, *69*, *74*
country codes, *16*

episodes_comments *(*media_comments*)*, 22
episodes_lists *(*media_lists*)*, 24
episodes_people *(*media_people*)*, 27
episodes_ratings *(*media_ratings*)*, 29
episodes_stats *(*media_stats*)*, 30
episodes_summary, 14
episodes_summary(), *11*, *24*, *26*, *28*, *30, 31*, *33*, *35*, *39*, *42*, *52*, *55, 56*, *58*, *79*
episodes_translations *(*media_translations*)*, 32
episodes_watching *(*media_watching*)*, 33

filters, *4, 5*, *8, 9*, 15, *20*, *44–46*, *48, 49*, *63*, *65, 66*, *85*, *87*
filters_episodes *(*filters*)*, 15
filters_episodes(), *4*, *8*, *44*, *46*, *49*, *63*, *85*
filters_movies *(*filters*)*, 15
filters_movies(), *4*, *8*, *44*, *46*, *49*, *63*, *85*
filters_shows *(*filters*)*, 15
filters_shows(), *4*, *8*, *44*, *46*, *49*, *63*, *85*
futurama, 18, *19*

gameofthrones, *18*, 19

genre slugs, *16*

httr2, *59*

ISOdate, *70*

language codes, *16*
lists_popular, 20
lists_popular(), *5*, *9*, *17*, *26*, *45*, *48*, *65, 66*, *72*, *74–76*, *87*
lists_trending *(*lists_popular*)*, 20
lubridate::as_date(), *61*
lubridate::ymd_hms(), *61*

media_aliases, *5*, *9*, 21, *23, 24*, *26*, *28*, *30, 31*, *33–39*, *41*, *45*, *48*, *56–58*, *65, 66*, *86*
media_comments, *5*, *9*, *11–13*, *15*, *21*, 22, *26*, *28*, *30, 31*, *33–39*, *41*, *45*, *48*, *52–58*, *65, 66*, *69*, *74*, *86*
media_lists, *5*, *9*, *15*, *20, 21*, *23*, 24, *24*, *28*, *30, 31*, *33–39*, *41, 42*, *45*, *48*, *52–58*, *65, 66*, *72*, *74–76*, *86*
media_people, *5*, *9*, *15*, *21*, *23, 24*, *26*, 27, *30, 31*, *33–39*, *41, 42*, *45*, *48*, *52–58*, *65, 66*, *86*
media_ratings, 29
media_ratings(), *5*, *9*, *15*, *21*, *23, 24*, *26*, *28*, *31*, *33–39*, *41*, *45*, *48*, *52–58*, *65, 66*, *86*
media_stats, 30
media_stats(), *5*, *9*, *15*, *21*, *23, 24*, *26*, *28*, *30*, *33–39*, *41*, *45*, *48*, *52–58*, *65, 66*, *86*
media_translations, *5*, *9*, *15*, *21*, *23, 24*, *26*, *28*, *30, 31*, 32, *34–39*, *41*, *45*, *48*, *52*, *55–58*, *65, 66*, *86*
media_watching, *5*, *9*, *15*, *21*, *23, 24*, *26*, *28*, *30, 31*, 33, *33*, *36–39*, *41*, *45*, *48*, *52*, *55–58*, *65, 66*, *86*
movies_aliases *(*media_aliases*)*, 21
movies_anticipated *(*anticipated_media*)*, 3
movies_boxoffice, 35
movies_boxoffice(), *5*, *9*, *21*, *23*, *26*, *28*, *30, 31*, *33, 34*, *37–39*, *41*, *45*, *48*, *65, 66*, *87*
movies_collected *(*collected_media*)*, 7
movies_comments *(*media_comments*)*, 22
movies_lists *(*media_lists*)*, 24
movies_people *(*media_people*)*, 27
movies_played *(*played_media*)*, 43
movies_popular *(*popular_media*)*, 45
movies_popular(), *15*
movies_ratings *(*media_ratings*)*, 29
movies_related, 36
movies_related(), *5*, *9*, *21*, *23*, *26*, *28*, *30, 31*, *33, 34*, *36*, *38, 39*, *41*, *45*, *48*, *65, 66*, *87*
movies_releases, 37
movies_releases(), *5*, *9*, *21*, *23*, *26*, *28*, *30, 31*, *33, 34*, *36, 37*, *39*, *41*, *45*, *48*, *65, 66*, *87*
movies_stats *(*media_stats*)*, 30
movies_summary, 38
movies_summary(), *5*, *9*, *11*, *15*, *21*, *23*, *26*, *28*, *30, 31*, *33, 34*, *36–38*, *41, 42*, *45*, *48*, *55*, *58*, *65, 66*, *79*, *87*
movies_translations *(*media_translations*)*, 32
movies_trending *(*trending_media*)*, 62
movies_updates *(*updated_media*)*, 65
movies_watched *(*watched_media*)*, 84
movies_watching *(*media_watching*)*, 33

pad_episode, 39
pad_episode(), *6*
people_lists *(*media_lists*)*, 24
people_media, *28*, 40
people_media(), *5*, *9*, *21*, *23, 24*, *26*, *28*, *30, 31*, *33–39*, *42*, *45*, *48*, *56–58*, *65, 66*, *87*
people_movies *(*people_media*)*, 40
people_shows *(*people_media*)*, 40
people_summary, 41
people_summary(), *11*, *15*, *26*, *28*, *39*, *41*, *55*, *58*, *79*
Played, *5*, *9*, *45*, *47*, *64*, *66*, *86*
played_media, *5*, *9*, *17*, *20, 21*, *23, 24*, *26*, *28*, *30, 31*, *33–39*, *41*, 43, *48*, *56–58*, *65, 66*, *87*
Popular, *5*, *9*, *45*, *47*, *64*, *66*, *86*
popular_media, *5*, *9*, *17*, *20, 21*, *23*, *26*, *28*, *30, 31*, *33, 34*, *36–39*, *41*, 45, *45*, *65, 66*, *87*

search_id *(*search_query*)*, 48
search_query, 48, *60*

search_query(), *15*
seasons_comments *(*media_comments*)*, 22
seasons_episodes, *29*, 51
seasons_episodes(), *14, 15*, *24*, *26*, *28*, *30, 31*, *33*, *35*, *53–56*
seasons_lists *(*media_lists*)*, 24
seasons_people *(*media_people*)*, 27
seasons_ratings *(*media_ratings*)*, 29
seasons_season, 52
seasons_season(), *24*, *26*, *28*, *30, 31*, *52*, *54*
seasons_stats *(*media_stats*)*, 30
seasons_summary, *51, 52*, 53, *53*
seasons_summary(), *11*, *14, 15*, *24*, *26*, *28*, *30, 31*, *33*, *35*, *39*, *42*, *52, 53*, *56*, *58*, *79*
seasons_watching *(*media_watching*)*, 33
shows_aliases *(*media_aliases*)*, 21
shows_anticipated *(*anticipated_media*)*, 3
shows_collected *(*collected_media*)*, 7
shows_comments *(*media_comments*)*, 22
shows_last_episode *(*shows_next_episode*)*, 55
shows_lists *(*media_lists*)*, 24
shows_next_episode, 55
shows_next_episode(), *9*, *15*, *21*, *24*, *26*, *28*, *30, 31*, *33*, *35*, *41*, *45*, *52*, *55*, *57, 58*, *66*
shows_people *(*media_people*)*, 27
shows_played *(*played_media*)*, 43
shows_popular, *60*
shows_popular *(*popular_media*)*, 45
shows_ratings *(*media_ratings*)*, 29
shows_related, 56
shows_related(), *9*, *21*, *24*, *26*, *28*, *30, 31*, *33*, *35*, *41*, *45*, *56*, *58*, *66*
shows_stats *(*media_stats*)*, 30
shows_summary, 57
shows_summary(), *9*, *11*, *15*, *21*, *24*, *26*, *28*, *30, 31*, *33*, *35*, *39*, *41, 42*, *45*, *55–57*, *66*, *79*
shows_translations *(*media_translations*)*, 32
shows_trending *(*trending_media*)*, 62
shows_trending(), *15*
shows_updates *(*updated_media*)*, 65
shows_watched *(*watched_media*)*, 84
shows_watching *(*media_watching*)*, 33

tibble, *50*
tibble(), *5*, *8*, *10*, *12–14*, *18–21*, *23*, *26*, *29*, *31, 32*, *34–38*, *42*, *44*, *47*, *51*, *53, 54*, *56–58*, *60, 61*, *64, 65*, *67*, *69–73*, *75–77*, *79, 80*, *82, 83*, *86*
tibbles, *28*, *41*, *81*
tidyr::unnest(), *67*
trakt_api_available, 59
trakt_api_available(), *61*
trakt_certifications, *5*, *8*, *44*, *47*, *50*, *64*, *86*
trakt_certifications *(*trakt_datasets*)*, 60
trakt_countries, *4*, *8*, *37*, *44*, *47*, *50*, *63*, *85*
trakt_countries *(*trakt_datasets*)*, 60
trakt_credentials, 59
trakt_datasets, 60
trakt_filters, *4*, *8*, *44*, *46*, *49*, *63*, *85*
trakt_genres, *4*, *8*, *44*, *47*, *50*, *63*, *85*
trakt_genres *(*trakt_datasets*)*, 60
trakt_get, *6*, 61
trakt_get(), *59*
trakt_languages, *4*, *8*, *32*, *44*, *47*, *50*, *63*, *85*
trakt_languages *(*trakt_datasets*)*, 60
trakt_networks, *5*, *8*, *17*, *44*, *47*, *50*, *64*, *86*
trakt_networks *(*trakt_datasets*)*, 60
tRakt_sitrep, 62
Trending, *5*, *9*, *45*, *47*, *64*, *66*, *86*
trending_media, *5*, *9*, *17*, *20, 21*, *23*, *26*, *28*, *30, 31*, *33, 34*, *36–39*, *41*, *45*, *48*, 62, *66*, *87*

updated_media, *5*, *9*, *17*, *20, 21*, *23, 24*, *26*, *28*, *30, 31*, *33–39*, *41*, *45*, *48*, *56–58*, 65, *65*, *87*
user_collection, 67
user_collection(), *69–71*, *78–81*, *83, 84*
user_comments, 68
user_comments(), *11–13*, *23*, *68*, *70, 71*, *74*, *78–81*, *83, 84*
user_followers *(*user_network*)*, 77
user_following *(*user_network*)*, 77
user_friends *(*user_network*)*, 77
user_history, 69
user_history(), *68, 69*, *71*, *78–81*, *83, 84*
user_likes, 71
user_likes(), *68–70*, *78–81*, *83, 84*
user_list, 72, *72*
user_list(), *20*, *26*, *74–76*
user_list_comments, 73

user_list_comments(), *11–13*, *20*, *23*, *26*, *69*, *72*, *75*, *76*
user_list_items, *72*, 74, *75*, *76*
user_list_items(), *20*, *26*, *72*, *74*, *76*
user_lists, *72–74*, 75, *76*
user_lists(), *20*, *26*, *72*, *74*, *75*
user_network, 77
user_network(), *68–71*, *79–81*, *83*, *84*
user_profile, *72*, *75*, 78
user_profile(), *11*, *15*, *39*, *42*, *55*, *58*, *68–71*, *78*, *80*, *81*, *83*, *84*
user_ratings, 79
user_ratings(), *68–71*, *78*, *79*, *81*, *83*, *84*
user_stats, 81
user_stats(), *68–71*, *78–80*, *83*, *84*
user_watched, 82
user_watched(), *68–71*, *78–81*, *84*
user_watchlist, 83
user_watchlist(), *68–71*, *78–81*, *83*

Watched, *5*, *9*, *45*, *48*, *64*, *66*, *86*
watched_media, *5*, *9*, *17*, *20*, *21*, *23*, *26*, *28*, *30*, *31*, *33*, *34*, *36–39*, *41*, *45*, *48*, *65*, *66*, 84