

# Package ‘tuber’

August 25, 2025

**Title** Client for the YouTube API

**Version** 1.1.0

**Maintainer** Gaurav Sood <gsood07@gmail.com>

**Description** Get comments posted on YouTube videos, information on how many times a video has been liked, search for videos with particular content, and much more. You can also scrape captions from a few videos. To learn more about the YouTube API, see <https://developers.google.com/youtube/v3/>.

**License** MIT + file LICENSE

**URL** <https://gojiplus.github.io/tuber/>,  
<https://github.com/gojiplus/tuber>

**BugReports** <https://github.com/gojiplus/tuber/issues>

**Depends** R (>= 4.1.0)

**Imports** askpass, dplyr, httr, httr2, jsonlite, magrittr, mime, plyr, purrr, tibble, tidyr, tidyselect, utils

**Suggests** hms, knitr (>= 1.11), lintr, rmarkdown, testthat, xml2

**VignetteBuilder** knitr

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**NeedsCompilation** no

**Author** Gaurav Sood [aut, cre],  
Kate Lyons [ctb],  
John Muschelli [ctb]

**Repository** CRAN

**Date/Publication** 2025-08-25 03:10:02 UTC

## Contents

|                                       |    |
|---------------------------------------|----|
| add_video_to_playlist . . . . .       | 3  |
| change_playlist_title . . . . .       | 4  |
| create_playlist . . . . .             | 4  |
| delete_captions . . . . .             | 5  |
| delete_channel_sections . . . . .     | 6  |
| delete_comments . . . . .             | 6  |
| delete_playlists . . . . .            | 7  |
| delete_playlist_items . . . . .       | 8  |
| delete_videos . . . . .               | 8  |
| get_all_channel_video_stats . . . . . | 9  |
| get_all_comments . . . . .            | 10 |
| get_captions . . . . .                | 11 |
| get_channel_stats . . . . .           | 12 |
| get_comments . . . . .                | 13 |
| get_comment_threads . . . . .         | 14 |
| get_playlists . . . . .               | 15 |
| get_playlist_items . . . . .          | 16 |
| get_playlist_item_ids . . . . .       | 18 |
| get_playlist_item_videoids . . . . .  | 19 |
| get_related_videos . . . . .          | 20 |
| get_stats . . . . .                   | 21 |
| get_subscriptions . . . . .           | 22 |
| get_video_details . . . . .           | 23 |
| list_abuse_report_reasons . . . . .   | 24 |
| list_captions . . . . .               | 25 |
| list_caption_tracks . . . . .         | 26 |
| list_channel_activities . . . . .     | 27 |
| list_channel_resources . . . . .      | 28 |
| list_channel_sections . . . . .       | 30 |
| list_channel_videos . . . . .         | 31 |
| list_guidecats . . . . .              | 32 |
| list_langs . . . . .                  | 33 |
| list_my_videos . . . . .              | 33 |
| list_regions . . . . .                | 34 |
| list_videocats . . . . .              | 35 |
| list_videos . . . . .                 | 36 |
| quota_management . . . . .            | 37 |
| read_sbv . . . . .                    | 37 |
| tuber . . . . .                       | 38 |
| tuber_check . . . . .                 | 38 |
| tuber_DELETE . . . . .                | 38 |
| tuber_GET . . . . .                   | 39 |
| tuber_POST . . . . .                  | 39 |
| tuber_POST_json . . . . .             | 40 |
| unicode_utils . . . . .               | 40 |
| update_video_metadata . . . . .       | 40 |

|                              |   |
|------------------------------|---|
| <i>add_video_to_playlist</i> | 3 |
|------------------------------|---|

|                              |    |
|------------------------------|----|
| upload_caption . . . . .     | 41 |
| upload_video . . . . .       | 42 |
| yt_get_quota_usage . . . . . | 43 |
| yt_key . . . . .             | 44 |
| yt_oauth . . . . .           | 45 |
| yt_reset_quota . . . . .     | 46 |
| yt_search . . . . .          | 47 |
| yt_set_quota_limit . . . . . | 49 |
| yt_token . . . . .           | 50 |
| yt_topic_search . . . . .    | 50 |

|              |           |
|--------------|-----------|
| <b>Index</b> | <b>52</b> |
|--------------|-----------|

---

|                       |                              |
|-----------------------|------------------------------|
| add_video_to_playlist | <i>Add Video to Playlist</i> |
|-----------------------|------------------------------|

---

**Description**

Add Video to Playlist

**Usage**

add\_video\_to\_playlist(playlist\_id, video\_id, position = NULL, ...)

**Arguments**

|             |                                                                                                                                    |
|-------------|------------------------------------------------------------------------------------------------------------------------------------|
| playlist_id | string; Required. The ID of the playlist.                                                                                          |
| video_id    | string; Required. The ID of the video to add.                                                                                      |
| position    | numeric; Optional. The position of the video in the playlist. If not provided, the video will be added to the end of the playlist. |
| ...         | Additional arguments passed to <a href="#">tuber_POST_json</a> .                                                                   |

**Value**

Details of the added video in the playlist.

**References**

<https://developers.google.com/youtube/v3/docs/playlistItems/insert>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

add_video_to_playlist(playlist_id = "YourPlaylistID", video_id = "2_gLD1jarfU")

## End(Not run)
```

---

`change_playlist_title` *Change the title of a YouTube playlist.*

---

### Description

This function updates the title of an existing YouTube playlist using the YouTube Data API.

### Usage

```
change_playlist_title(playlist_id, new_title)
```

### Arguments

|                          |                                                                          |
|--------------------------|--------------------------------------------------------------------------|
| <code>playlist_id</code> | A character string specifying the ID of the playlist you want to update. |
| <code>new_title</code>   | A character string specifying the new title for the playlist.            |

### Value

A list containing the server response after the update attempt.

### Examples

```
## Not run:  
change_playlist_title(playlist_id = "YourPlaylistID", new_title = "New Playlist Title")  
  
## End(Not run)
```

---

`create_playlist` *Create New Playlist*

---

### Description

Create New Playlist

### Usage

```
create_playlist(title, description, status, ...)
```

### Arguments

|                          |                                                                                          |
|--------------------------|------------------------------------------------------------------------------------------|
| <code>title</code>       | string; Required. The title of the playlist.                                             |
| <code>description</code> | string; Optional. The description of the playlist.                                       |
| <code>status</code>      | string; Optional. Default: 'public'. Can be one of: 'private', 'public', or 'un-listed'. |
| <code>...</code>         | Additional arguments passed to <a href="#">tuber_POST</a> .                              |

**Value**

The created playlist's details.

**References**

<https://developers.google.com/youtube/v3/docs/playlists/insert>

**Examples**

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
create_playlist(title = "My New Playlist", description = "This is a test playlist.")  
  
## End(Not run)
```

---

|                 |                                          |
|-----------------|------------------------------------------|
| delete_captions | <i>Delete a Particular Caption Track</i> |
|-----------------|------------------------------------------|

---

**Description**

Delete a Particular Caption Track

**Usage**

```
delete_captions(id = NULL, ...)
```

**Arguments**

|     |                                                                   |
|-----|-------------------------------------------------------------------|
| id  | String. Required. id of the caption track that is being retrieved |
| ... | Additional arguments passed to <code>tuber_DELETE</code> .        |

**References**

<https://developers.google.com/youtube/v3/docs/captions/delete>

**Examples**

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
delete_captions(id = "y3ElXcEME3lSISz6izkWWT5GvxjPu8pA")  
  
## End(Not run)
```

---

`delete_channel_sections`*Delete Channel Sections*

---

**Description**

Delete a Channel Section

**Usage**

```
delete_channel_sections(id = NULL, ...)
```

**Arguments**

|                  |                                                               |
|------------------|---------------------------------------------------------------|
| <code>id</code>  | Required. ID of the channel section.                          |
| <code>...</code> | Additional arguments passed to <a href="#">tuber_DELETE</a> . |

**References**

<https://developers.google.com/youtube/v3/docs/channelSections/delete>

**Examples**

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
delete_channel_sections(c(channel_id = "UCRw8bIz2wMLmfgAgWm903cA"))  
  
## End(Not run)
```

---

`delete_comments`*Delete a Particular Comment*

---

**Description**

Delete a Particular Comment

**Usage**

```
delete_comments(id = NULL, ...)
```

**Arguments**

|                  |                                                               |
|------------------|---------------------------------------------------------------|
| <code>id</code>  | String. Required. id of the comment being retrieved           |
| <code>...</code> | Additional arguments passed to <a href="#">tuber_DELETE</a> . |

## References

<https://developers.google.com/youtube/v3/docs/comments/delete>

## Examples

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
delete_comments(id = "y3ElXcEME3lSISz6izkWVT5GvxjPu8pA")  
  
## End(Not run)
```

---

|                  |                          |
|------------------|--------------------------|
| delete_playlists | <i>Delete a Playlist</i> |
|------------------|--------------------------|

---

## Description

Delete a Playlist

## Usage

```
delete_playlists(id = NULL, ...)
```

## Arguments

|     |                                                            |
|-----|------------------------------------------------------------|
| id  | String. Required. id of the playlist that is to be deleted |
| ... | Additional arguments passed to <code>tuber_DELETE</code> . |

## References

<https://developers.google.com/youtube/v3/docs/playlists/delete>

## Examples

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
delete_playlists(id = "y3ElXcEME3lSISz6izkWVT5GvxjPu8pA")  
  
## End(Not run)
```

---

|                       |                               |
|-----------------------|-------------------------------|
| delete_playlist_items | <i>Delete a Playlist Item</i> |
|-----------------------|-------------------------------|

---

**Description**

Delete a Playlist Item

**Usage**

```
delete_playlist_items(id = NULL, ...)
```

**Arguments**

|     |                                                                 |
|-----|-----------------------------------------------------------------|
| id  | String. Required. id of the playlist item that is to be deleted |
| ... | Additional arguments passed to <a href="#">tuber_DELETE</a> .   |

**References**

<https://developers.google.com/youtube/v3/docs/playlistItems/delete>

**Examples**

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
delete_playlist_items(id = "YourPlaylistItemID")  
  
## End(Not run)
```

---

|               |                       |
|---------------|-----------------------|
| delete_videos | <i>Delete a Video</i> |
|---------------|-----------------------|

---

**Description**

Delete a Video

**Usage**

```
delete_videos(id = NULL, ...)
```

**Arguments**

|     |                                                               |
|-----|---------------------------------------------------------------|
| id  | String. Required. id of the video that is to be deleted       |
| ... | Additional arguments passed to <a href="#">tuber_DELETE</a> . |

## References

<https://developers.google.com/youtube/v3/docs/playlistItems/delete>

## Examples

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
delete_videos(id = "y3ElXcEME3lSISz6izkWVT5GvxjPu8pA")  
  
## End(Not run)
```

---

get\_all\_channel\_video\_stats

*Get statistics on all the videos in a Channel*

---

## Description

Iterates through the channel's uploads playlist, collecting video IDs from each page until no further pages are available. Statistics and details are then fetched for every video.

## Usage

```
get_all_channel_video_stats(channel_id = NULL, mine = FALSE, ...)
```

## Arguments

|            |                                                                                 |
|------------|---------------------------------------------------------------------------------|
| channel_id | Character. Id of the channel                                                    |
| mine       | Boolean. TRUE if you want to fetch stats of your own channel. Default is FALSE. |
| ...        | Additional arguments passed to <a href="#">tuber_GET</a> .                      |

## Value

A data.frame containing video metadata along with view, like, dislike and comment counts.

If the channel\_id is mistyped or there is no information, an empty list is returned

## References

<https://developers.google.com/youtube/v3/docs/channels/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

get_all_channel_video_stats(channel_id="UCx0hDvtaoXDAB336Ao1Ws3A")
get_all_channel_video_stats(channel_id="UCMtFAi84ehTSYSE9Xo") # Incorrect channel ID

## End(Not run)
```

---

|                  |                                                           |
|------------------|-----------------------------------------------------------|
| get_all_comments | <i>Get all the comments for a video including replies</i> |
|------------------|-----------------------------------------------------------|

---

**Description**

Get all the comments for a video including replies

**Usage**

```
get_all_comments(video_id = NULL, ...)
```

**Arguments**

|          |                                                            |
|----------|------------------------------------------------------------|
| video_id | string; Required. video_id: video ID.                      |
| ...      | Additional arguments passed to <a href="#">tuber_GET</a> . |

**Value**

a data.frame with the following columns: authorDisplayName, authorProfileImageUrl, authorChannelUrl, authorChannelId.value, videoId, textDisplay, canRate, viewerRating, likeCount, publishedAt, updatedAt, id, moderationStatus, parentId

**References**

<https://developers.google.com/youtube/v3/docs/commentThreads/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

get_all_comments(video_id = "a-UQz7fqR3w")

## End(Not run)
```

---

`get_captions`*Get Particular Caption Track*

---

### Description

For getting captions from the v3 API, you must specify the id resource. Check [list\\_caption\\_tracks](#) for more information.

### Usage

```
get_captions(id = NULL, lang = "en", format = "sbv", as_raw = TRUE, ...)
```

### Arguments

|                     |                                                                              |
|---------------------|------------------------------------------------------------------------------|
| <code>id</code>     | String. Required. id of the caption track that is being retrieved            |
| <code>lang</code>   | Optional. Default is en.                                                     |
| <code>format</code> | Optional. Default is sbv.                                                    |
| <code>as_raw</code> | If FALSE the captions be converted to a character string versus a raw vector |
| <code>...</code>    | Additional arguments passed to <a href="#">tuber_GET</a> .                   |

### Value

String.

### References

<https://developers.google.com/youtube/v3/docs/captions/download>

### Examples

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
get_captions(id = "y3E1XcEME3lSISz6izkWVT5GvxjPu8pA")  
  
## End(Not run)
```

---

|                   |                                    |
|-------------------|------------------------------------|
| get_channel_stats | <i>Get statistics of a Channel</i> |
|-------------------|------------------------------------|

---

## Description

Get statistics of a Channel

## Usage

```
get_channel_stats(channel_id = NULL, mine = NULL, ...)

list_my_channel(...)
```

## Arguments

|            |                                                                                |
|------------|--------------------------------------------------------------------------------|
| channel_id | Character. Id of the channel                                                   |
| mine       | Boolean. TRUE if you want to fetch stats of your own channel. Default is NULL. |
| ...        | Additional arguments passed to <a href="#">tuber_GET</a> .                     |

## Value

nested named list with top element names: kind, etag, id, snippet (list of details of the channel including title), statistics (list of 5)

If the channel\_id is mistyped or there is no information, an empty list is returned

## References

<https://developers.google.com/youtube/v3/docs/channels/list>

## Examples

```
## Not run:

# Set API token via yt_oauth() first

get_channel_stats(channel_id="UCMtFAi84ehTSYSE9XoHefig")
get_channel_stats(channel_id="UCMtFAi84ehTSYSE9Xo") # Incorrect channel ID

## End(Not run)
```

get\_comments

*Get Comments***Description**

Get Comments

**Usage**

```
get_comments(
  filter = NULL,
  part = "snippet",
  max_results = 100,
  text_format = "html",
  page_token = NULL,
  simplify = TRUE,
  ...
)
```

**Arguments**

|             |                                                                                                                                                            |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| filter      | string; Required. named vector of length 1 potential names of the entry in the vector: comment_id: comment ID. parent_id: parent ID.                       |
| part        | Comment resource requested. Required. Comma separated list of one or more of the following: id, snippet. e.g., "id, snippet", "id", etc. Default: snippet. |
| max_results | Maximum number of items that should be returned. Integer. Optional. Can be between 20 and 100. Default is 100.                                             |
| text_format | Data Type: Character. Default is "html". Only takes "html" or "plainText". Optional.                                                                       |
| page_token  | Specific page in the result set that should be returned. Optional.                                                                                         |
| simplify    | Data Type: Boolean. Default is TRUE. If TRUE, the function returns a data frame. Else a list with all the information returned.                            |
| ...         | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                 |

**Value**

Nested named list. The entry items is a list of comments along with meta information. Within each of the items is an item snippet which has an item topLevelComment\$snippet\$textDisplay that contains the actual comment.

When filter is comment\_id, and simplify is TRUE, and there is a correct comment id, it returns a data.frame with the following cols: id, authorDisplayName, authorProfileImageUrl, authorChannelUrl, value, textDisplay, canRate, viewerRating, likeCount publishedAt, updatedAt

**References**

<https://developers.google.com/youtube/v3/docs/comments/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

get_comments(filter = c(comment_id = "z13dh13j5rr0wbmq04cifrhtuypw14hsdk"))
get_comments(filter = c(parent_id = "z13ds5yxjq3zzptyx04chlkbhx2yh3ezxtc0k"))
get_comments(filter =
c(comment_id = "z13dh13j5rr0wbmq04cifrhtuypw14hsdk,
      z13dh13j5rr0wbmq04cifrhtuypw14hsdk"))

## End(Not run)
```

---

|                     |                             |
|---------------------|-----------------------------|
| get_comment_threads | <i>Get Comments Threads</i> |
|---------------------|-----------------------------|

---

**Description**

Get Comments Threads

**Usage**

```
get_comment_threads(
  filter = NULL,
  part = "snippet",
  text_format = "html",
  simplify = TRUE,
  max_results = 100,
  page_token = NULL,
  ...
)
```

**Arguments**

|             |                                                                                                                                                                                                                                  |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| filter      | string; Required. named vector of length 1 potential names of the entry in the vector: video_id: video ID. channel_id: channel ID. thread_id: comma-separated list of comment thread IDs threads_related_to_channel: channel ID. |
| part        | Comment resource requested. Required. Comma separated list of one or more of the following: id, snippet. e.g., "id, snippet", "id", etc. Default: snippet.                                                                       |
| text_format | Data Type: Character. Default is "html". Only takes "html" or "plainText". Optional.                                                                                                                                             |
| simplify    | Data Type: Boolean. Default is TRUE. If TRUE, the function returns a data frame. Else a list with all the information returned.                                                                                                  |
| max_results | Maximum number of items that should be returned. Integer. Optional. Default is 100. If the value is greater than 100 then the function fetches all the results. The outcome is a simplified data.frame.                          |

page\_token      Specific page in the result set that should be returned. Optional.  
 ...              Additional arguments passed to [tuber\\_GET](#).

### Value

Nested named list. The entry items is a list of comments along with meta information. Within each of the items is an item snippet which has an item topLevelComment\$snippet\$textDisplay that contains the actual comment.

If simplify is TRUE, a data.frame with the following columns: authorDisplayName, authorProfileImageUrl, authorChannelUrl, authorChannelId.value, videoId, textDisplay, canRate, viewerRating, likeCount, publishedAt, updatedAt

### References

<https://developers.google.com/youtube/v3/docs/commentThreads/list>

### Examples

```
## Not run:

# Set API token via yt_oauth() first

get_comment_threads(filter = c(video_id = "N708P-A45D0"))
get_comment_threads(filter = c(video_id = "N708P-A45D0"), max_results = 101)

## End(Not run)
```

---

|               |                      |
|---------------|----------------------|
| get_playlists | <i>Get Playlists</i> |
|---------------|----------------------|

---

### Description

Get Playlists

### Usage

```
get_playlists(
  filter = NULL,
  part = "snippet",
  max_results = 50,
  hl = NULL,
  page_token = NULL,
  simplify = TRUE,
  ...
)
```

**Arguments**

|             |                                                                                                                                                                         |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| filter      | string; Required. named vector of length 1 potential names of the entry in the vector: channel_id: ID of the channel playlist_id: YouTube playlist ID.                  |
| part        | Required. One of the following: contentDetails, id,localizations, player, snippet, status. Default: contentDetails.                                                     |
| max_results | Maximum number of items that should be returned. Integer. Optional. Default is 50. Values over 50 trigger additional requests and may increase API quota usage.         |
| hl          | Language used for text values. Optional. Default is en-US. For other allowed language codes, see <a href="#">list_langs</a> .                                           |
| page_token  | specific page in the result set that should be returned, optional                                                                                                       |
| simplify    | Data Type: Boolean. Default is TRUE. If TRUE and if part requested is contentDetails, the function returns a data.frame. Else a list with all the information returned. |
| ...         | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                              |

**Value**

playlists When simplify is TRUE, a data.frame with 4 columns is returned: kind, etag, id, contentDetails.itemCount

**References**

<https://developers.google.com/youtube/v3/docs/playlists/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

get_playlists(filter=c(channel_id="UCMtFAi84ehTSYSE9XoHefig"))
get_playlists(filter=c(channel_id="UCMtFAi84ehTSYSE9X")) # incorrect Channel ID

## End(Not run)
```

---

|                    |                           |
|--------------------|---------------------------|
| get_playlist_items | <i>Get Playlist Items</i> |
|--------------------|---------------------------|

---

**Description**

Get Playlist Items

**Usage**

```
get_playlist_items(
  filter = NULL,
  part = "contentDetails",
  max_results = 50,
  video_id = NULL,
  page_token = NULL,
  simplify = TRUE,
  ...
)
```

**Arguments**

|             |                                                                                                                                                                                                                  |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| filter      | string; Required. named vector of length 1 potential names of the entry in the vector: item_id: comma-separated list of one or more unique playlist item IDs. playlist_id: YouTube playlist ID.                  |
| part        | Required. Comma separated string including one or more of the following: contentDetails, id, snippet, status. Default: contentDetails.                                                                           |
| max_results | Maximum number of items that should be returned. Integer. Optional. Default is 50. If over 50, additional requests are made until the requested amount is retrieved. Larger values may increase API quota usage. |
| video_id    | Optional. request should return only the playlist items that contain the specified video.                                                                                                                        |
| page_token  | specific page in the result set that should be returned, optional                                                                                                                                                |
| simplify    | returns a data.frame rather than a list.                                                                                                                                                                         |
| ...         | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                                                                       |

**Value**

playlist items

**References**

<https://developers.google.com/youtube/v3/docs/playlists/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

get_playlist_items(filter =
  c(playlist_id = "PLrEnWoR732-CN09YykVof2lxdI3ML0Zda"))
get_playlist_items(filter =
  c(playlist_id = "PL0f01XVeVW9QM03GoESky4yDgQfK2SsXN"),
  max_results = 51)

## End(Not run)
```

---

`get_playlist_item_ids` *Get Playlist Item IDs*

---

## Description

Get Playlist Item IDs

## Usage

```
get_playlist_item_ids(  
  filter = NULL,  
  part = "contentDetails",  
  max_results = 50,  
  video_id = NULL,  
  page_token = NULL,  
  simplify = TRUE,  
  ...  
)
```

## Arguments

|                          |                                                                                                                                                                                                                             |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>filter</code>      | string; Required. named vector of length 1 potential names of the entry in the vector: <code>item_id</code> : comma-separated list of one or more unique playlist item IDs. <code>playlist_id</code> : YouTube playlist ID. |
| <code>part</code>        | Required. Comma separated string including one or more of the following: <code>contentDetails</code> , <code>id</code> , <code>snippet</code> , <code>status</code> . Default: <code>contentDetails</code> .                |
| <code>max_results</code> | Maximum number of items that should be returned. Integer. Optional. Default is 50. Values over 50 will trigger multiple requests and may increase API quota usage.                                                          |
| <code>video_id</code>    | Optional. request should return only the playlist items that contain the specified video.                                                                                                                                   |
| <code>page_token</code>  | specific page in the result set that should be returned, optional                                                                                                                                                           |
| <code>simplify</code>    | returns a <code>data.frame</code> rather than a list.                                                                                                                                                                       |
| <code>...</code>         | Additional arguments passed to <code>tuber_GET</code> .                                                                                                                                                                     |

## Value

playlist items

## References

<https://developers.google.com/youtube/v3/docs/playlists/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

get_playlist_items(filter =
  c(playlist_id = "PLrEnWoR732-CN09YykVof2lxdI3MLOZda"))
get_playlist_items(filter =
  c(playlist_id = "PL0f01XVeVW9QM03GoESky4yDgQfK2SsXN"),
  max_results = 51)

## End(Not run)
```

---

```
get_playlist_item_videoids
  Get Playlist Item Video IDs
```

---

**Description**

Get Playlist Item Video IDs

**Usage**

```
get_playlist_item_videoids(
  filter = NULL,
  part = "contentDetails",
  max_results = 50,
  video_id = NULL,
  page_token = NULL,
  simplify = TRUE,
  ...
)
```

**Arguments**

|             |                                                                                                                                                                                                 |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| filter      | string; Required. named vector of length 1 potential names of the entry in the vector: item_id: comma-separated list of one or more unique playlist item IDs. playlist_id: YouTube playlist ID. |
| part        | Required. Comma separated string including one or more of the following: contentDetails, id, snippet, status. Default: contentDetails.                                                          |
| max_results | Maximum number of items that should be returned. Integer. Optional. Default is 50. Values over 50 trigger multiple requests and may increase API quota usage.                                   |
| video_id    | Optional. request should return only the playlist items that contain the specified video.                                                                                                       |
| page_token  | specific page in the result set that should be returned, optional                                                                                                                               |
| simplify    | returns a data.frame rather than a list.                                                                                                                                                        |
| ...         | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                                                      |

Value

playlist items

References

<https://developers.google.com/youtube/v3/docs/playlists/list>

Examples

```
## Not run:

# Set API token via yt_oauth() first

get_playlist_items(filter =
  c(playlist_id = "YourPlaylistID"))
get_playlist_items(filter =
  c(playlist_id = "YourPlaylistID"),
  max_results = 51)

## End(Not run)
```

---

|                    |                           |
|--------------------|---------------------------|
| get_related_videos | <i>Get Related Videos</i> |
|--------------------|---------------------------|

---

Description

Takes a video id and returns related videos

Usage

```
get_related_videos(
  video_id = NULL,
  max_results = 50,
  safe_search = "none",
  ...
)
```

Arguments

|             |                                                                                                                                                               |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| video_id    | Character. Required. No default.                                                                                                                              |
| max_results | Maximum number of items that should be returned. Integer. Optional. Default is 50. Values over 50 trigger multiple requests and may increase API quota usage. |
| safe_search | Character. Optional. Takes one of three values: 'moderate', 'none' (default) or 'strict'                                                                      |
| ...         | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                    |

Value

data.frame with 16 columns: video\_id, rel\_video\_id, publishedAt, channelId, title, description, thumbnails, thumbnails.default.width, thumbnails.default.height, thumbnails.medium.url, thumbnails.medium.width, thumbnails.medium.height, thumbnails.high.url, thumbnails.high.width, thumbnails.high.height, channelTitle, liveBroadcastContent

References

<https://developers.google.com/youtube/v3/docs/search/list>

Examples

```
## Not run:

# Set API token via yt_oauth() first

get_related_videos(video_id = "yJXTXN4xrI8")

## End(Not run)
```

---

|           |                                  |
|-----------|----------------------------------|
| get_stats | <i>Get statistics of a Video</i> |
|-----------|----------------------------------|

---

Description

Get statistics of a Video

Usage

```
get_stats(video_id = NULL, ...)
```

Arguments

- video\_id            Character. Id of the video. Required.
- ...                Additional arguments passed to [tuber\\_GET](#).

Value

list with 6 elements: id, viewCount, likeCount, dislikeCount, favoriteCount, commentCount

References

<https://developers.google.com/youtube/v3/docs/videos/list#parameters>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

get_stats(video_id="N708P-A45D0")

## End(Not run)
```

---

|                   |                          |
|-------------------|--------------------------|
| get_subscriptions | <i>Get Subscriptions</i> |
|-------------------|--------------------------|

---

**Description**

Get Subscriptions

**Usage**

```
get_subscriptions(
  filter = NULL,
  part = "contentDetails",
  max_results = 50,
  for_channel_id = NULL,
  order = NULL,
  page_token = NULL,
  ...
)
```

**Arguments**

|                |                                                                                                                                                                                                                      |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| filter         | string; Required. named vector of length 1 potential names of the entry in the vector: channel_id: ID of the channel. Required. No default. subscription_id: YouTube subscription ID                                 |
| part           | Part of the resource requested. Required. Character. A comma separated list of one or more of the following: contentDetails, id, snippet, subscriberSnippet. e.g. "id, snippet", "id", etc. Default: contentDetails. |
| max_results    | Maximum number of items that should be returned. Integer. Optional. Default is 50. Values over 50 will trigger additional requests and may increase API quota usage.                                                 |
| for_channel_id | Optional. String. A comma-separated list of channel IDs. Limits response to subscriptions matching those channels.                                                                                                   |
| order          | method that will be used to sort resources in the API response. Takes one of the following: alphabetical, relevance, unread                                                                                          |
| page_token     | Specific page in the result set that should be returned. Optional. String.                                                                                                                                           |
| ...            | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                                                                           |

**Value**

named list of subscriptions

**References**

<https://developers.google.com/youtube/v3/docs/subscriptions/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

get_subscriptions(filter = c(channel_id = "UChTJTbr5kf3hYazJZ0-euHg"))

## End(Not run)
```

---

|                   |                                         |
|-------------------|-----------------------------------------|
| get_video_details | <i>Get Details of a Video or Videos</i> |
|-------------------|-----------------------------------------|

---

**Description**

Get details such as when the video was published, the title, description, thumbnails, category etc.

**Usage**

```
get_video_details(
  video_id = NULL,
  part = "snippet",
  as.data.frame = FALSE,
  ...
)
```

**Arguments**

|               |                                                                                                                                                                                                                                                           |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| video_id      | Comma separated list of IDs of the videos for which details are requested. Required.                                                                                                                                                                      |
| part          | Comma-separated vector of video resource properties requested. Options include: contentDetails, fileDetails, id, liveStreamingDetails, localizations, player, processingDetails, recordingDetails, snippet, statistics, status, suggestions, topicDetails |
| as.data.frame | Logical, returns the requested information as data.frame. Does not work for: fileDetails, suggestions, processingDetails                                                                                                                                  |
| ...           | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                                                                                                                |

**Value**

list. If part is snippet, the list will have the following elements: id (video id that was passed), publishedAt, channelId, title, description, thumbnails, channelTitle, categoryId, liveBroadcastContent, localized, defaultAudioLanguage

**References**

<https://developers.google.com/youtube/v3/docs/videos/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

get_video_details(video_id = "yJXTXN4xrI8")
get_video_details(video_id = "yJXTXN4xrI8", part = "contentDetails")
# retrieve multiple parameters
get_video_details(video_id = "yJXTXN4xrI8", part = c("contentDetails", "status"))
# get details for multiple videos as data frame
get_video_details(video_id = c("LDZX4ooRsWs", "yJXTXN4xrI8"), as.data.frame = TRUE)

## End(Not run)
```

---

list\_abuse\_report\_reasons

*List reasons that can be used to report abusive videos*

---

**Description**

List reasons that can be used to report abusive videos

**Usage**

```
list_abuse_report_reasons(part = "id, snippet", hl = "en-US", ...)
```

**Arguments**

|      |                                                                                                                                                            |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| part | Caption resource requested. Required. Comma separated list of one or more of the following: id, snippet. e.g., "id, snippet", "id", etc. Default: snippet. |
| hl   | Language used for text values. Optional. Default is en-US. For other allowed language codes, see <a href="#">list_langs</a> .                              |
| ...  | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                 |

**Value**

If no results, empty data.frame returned If part requested = "id, snippet" or "snippet", data.frame with 4 columns: etag, id, label, secReasons If part requested = "id", data.frame with 2 columns: etag, id

**References**

<https://developers.google.com/youtube/v3/docs/videoAbuseReportReasons/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

list_abuse_report_reasons()
list_abuse_report_reasons(part="id")
list_abuse_report_reasons(part="snippet")

## End(Not run)
```

---

list\_captions

---

*Upload Video to Youtube*


---

**Description**

Upload Video to Youtube

**Usage**

```
list_captions(video_id, query = NULL, ...)
```

**Arguments**

|          |                                                                                             |
|----------|---------------------------------------------------------------------------------------------|
| video_id | ID of the YouTube video                                                                     |
| query    | Fields for 'query' in 'GET'                                                                 |
| ...      | Additional arguments to send to <a href="#">tuber_GET</a> and therefore <a href="#">GET</a> |

**Value**

A list of the response object from the GET and content about captions

**Examples**

```
## Not run:
video_id <- "M7FIvfx5J10"
list_captions(video_id)

## End(Not run)
```

---

list\_caption\_tracks     *List Captions of a Video*

---

## Description

List Captions of a Video

## Usage

```
list_caption_tracks(
  part = "snippet",
  video_id = NULL,
  lang = "en",
  id = NULL,
  simplify = TRUE,
  ...
)
```

## Arguments

|          |                                                                                                                                                      |
|----------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| part     | Caption resource requested. Required. Comma separated list of one or more of the following: id, snippet. e.g., "id, snippet", "id" Default: snippet. |
| video_id | ID of the video whose captions are requested. Required. No default.                                                                                  |
| lang     | Language of the caption; required; default is English ("en")                                                                                         |
| id       | comma-separated list of IDs that identify the caption resources that should be retrieved; optional; string                                           |
| simplify | Boolean. Default is TRUE. When TRUE, and part is snippet, a data.frame is returned                                                                   |
| ...      | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                           |

## Value

list of caption tracks. When simplify is TRUE, a data.frame is returned with following columns: videoId, lastUpdated, trackKind, language, name, audioTrackType, isCC, isLarge, isEasyReader, isDraft, isAutoSynced, status, id (caption id)

## References

<https://developers.google.com/youtube/v3/docs/captions/list>

## Examples

```
## Not run:

# Set API token via yt_oauth() first
```

```
list_caption_tracks(video_id = "yJXTXN4xrI8")

## End(Not run)
```

---

```
list_channel_activities
```

*List Channel Activity*

---

## Description

Returns a list of channel events that match the request criteria.

## Usage

```
list_channel_activities(
  filter = NULL,
  part = "snippet",
  max_results = 50,
  page_token = NULL,
  published_after = NULL,
  published_before = NULL,
  region_code = NULL,
  simplify = TRUE,
  ...
)
```

## Arguments

|                  |                                                                                                                                                                         |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| filter           | string; Required. named vector of length 1 potential names of the entry in the vector: channel_id: ID of the channel. Required. No default.                             |
| part             | specify which part do you want. It can only be one of the three: contentDetails, id, snippet. Default is snippet.                                                       |
| max_results      | Maximum number of items that should be returned. Integer. Optional. Default is 50. Values over 50 will trigger additional requests and may increase API quota usage.    |
| page_token       | specific page in the result set that should be returned, optional                                                                                                       |
| published_after  | Character. Optional. RFC 339 Format. For instance, "1970-01-01T00:00:00Z"                                                                                               |
| published_before | Character. Optional. RFC 339 Format. For instance, "1970-01-01T00:00:00Z"                                                                                               |
| region_code      | ISO 3166-1 alpha-2 country code, optional, see also <a href="#">list_regions</a>                                                                                        |
| simplify         | Data Type: Boolean. Default is TRUE. If TRUE and if part requested is contentDetails, the function returns a data.frame. Else a list with all the information returned. |
| ...              | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                              |

**Value**

named list If simplify is TRUE, a data.frame is returned with 18 columns: publishedAt, channelId, title, description, thumbnails.default.url, thumbnails.default.width, thumbnails.default.height, thumbnails.medium.width, thumbnails.medium.height, thumbnails.high.url, thumbnails.high.width, thumbnails.standard.url, thumbnails.standard.width, thumbnails.standard.height, channelTitle, type

**References**

<https://developers.google.com/youtube/v3/docs/activities/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

list_channel_activities(filter = c(channel_id = "UCRw8bIz2wMLmfgAgWm903cA"))
list_channel_activities(filter = c(channel_id = "UCRw8bIz2wMLmfgAgWm903cA", regionCode="US"))
list_channel_activities(filter = c(channel_id = "UCMtFAi84ehTSYSE9XoHefig"),
                        published_before = "2016-02-10T00:00:00Z",
                        published_after = "2016-01-01T00:00:00Z")

## End(Not run)
```

---

list\_channel\_resources

*Returns List of Requested Channel Resources*

---

**Description**

Returns List of Requested Channel Resources

**Usage**

```
list_channel_resources(
  filter = NULL,
  part = "contentDetails",
  max_results = 50,
  page_token = NULL,
  hl = "en-US",
  ...
)
```

**Arguments**

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| filter      | string; Required. named vector with a single valid name potential names of the entry in the vector: category_id: YouTube guide category that returns channels associated with that category username: YouTube username that returns the channel associated with that username. Multiple usernames can be provided. channel_id: a comma-separated list of the YouTube channel ID(s) for the resource(s) that are being retrieved |
| part        | a comma-separated list of channel resource properties that response will include a string. Required. One of the following: auditDetails, brandingSettings, contentDetails, contentOwnerDetails, id, invideoPromotion, localizations, snippet, statistics, status, topicDetails. Default is contentDetails.                                                                                                                      |
| max_results | Maximum number of items that should be returned. Integer. Optional. Default is 50. Values over 50 will trigger additional requests and may increase API quota usage.                                                                                                                                                                                                                                                            |
| page_token  | specific page in the result set that should be returned, optional                                                                                                                                                                                                                                                                                                                                                               |
| hl          | Language used for text values. Optional. The default is en-US. For other allowed language codes, see <a href="#">list_langs</a> .                                                                                                                                                                                                                                                                                               |
| ...         | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                                                                                                                                                                                                                                                                                      |

**Value**

list. If username is used in filter, a data frame with columns username and channel\_id is returned.

**References**

<https://developers.google.com/youtube/v3/docs/channels/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

list_channel_resources(filter = c(channel_id = "UCT5Cx1l4IS3wHkJXNyuj4TA"))
list_channel_resources(filter = c(username = "latenight"), part = "id")
list_channel_resources(filter = c(username = c("latenight", "PBS")),
                        part = "id")

## End(Not run)
```

---

**list\_channel\_sections** *List Channel Sections*

---

**Description**

Returns list of channel sections that channel id belongs to.

**Usage**

```
list_channel_sections(filter = NULL, part = "snippet", hl = NULL, ...)
```

**Arguments**

|        |                                                                                                                                                 |
|--------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| filter | string; Required. named vector of length 1 potential names of the entry in the vector: channel_id: Channel ID id: Section ID                    |
| part   | specify which part do you want. It can only be one of the following: contentDetails, id, localizations, snippet, targeting. Default is snippet. |
| hl     | language that will be used for text values, optional, default is en-US. See also <a href="#">list_langs</a>                                     |
| ...    | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                      |

**Value**

captions for the video from one of the first track

**References**

<https://developers.google.com/youtube/v3/docs/activities/list>

**Examples**

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
list_channel_sections(c(channel_id = "UCRw8bIz2wMLmfgAgWm903cA"))  
  
## End(Not run)
```

---

list\_channel\_videos      *Returns List of Requested Channel Videos*

---

### Description

Iterate through the max\_results number of playlists in channel and get the videos for each of the playlists.

### Usage

```
list_channel_videos(
  channel_id = NULL,
  max_results = 50,
  page_token = NULL,
  hl = "en-US",
  ...
)
```

### Arguments

|             |                                                                                                                             |
|-------------|-----------------------------------------------------------------------------------------------------------------------------|
| channel_id  | String. ID of the channel. Required.                                                                                        |
| max_results | Maximum number of videos returned. Integer. Default is 50. If the number is over 50, all the videos will be returned.       |
| page_token  | Specific page in the result set that should be returned. Optional.                                                          |
| hl          | Language used for text values. Optional. Default is en-US. For other allowed language codes, see <a href="#">list_langs</a> |
| ...         | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                  |

### Value

list of data.frame with each list corresponding to a different playlist

### References

<https://developers.google.com/youtube/v3/docs/channels/list>

### Examples

```
## Not run:

# Set API token via yt_oauth() first

list_channel_videos(channel_id = "UCX0KEdf0FxshO_-Su3K8SHg")
list_channel_videos(channel_id = "UCX0KEdf0FxshO_-Su3K8SHg", max_results = 10)

## End(Not run)
```

---

|                |                                                                            |
|----------------|----------------------------------------------------------------------------|
| list_guidecats | <i>Get list of categories that can be associated with YouTube channels</i> |
|----------------|----------------------------------------------------------------------------|

---

### Description

Get list of categories that can be associated with YouTube channels

### Usage

```
list_guidecats(filter = NULL, hl = NULL, ...)
```

### Arguments

|        |                                                                                                                                                                                                                                                                                                  |
|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| filter | string; Required. named vector of length 1 potential names of the entry in the vector: region_code: Character. Required. Has to be a ISO 3166-1 alpha-2 code (see <a href="https://www.iso.org/obp/ui/#search">https://www.iso.org/obp/ui/#search</a> ) category_id: YouTube channel category ID |
| hl     | Language used for text values. Optional. Default is en-US. For other allowed language codes, see <a href="#">list_langs</a> .                                                                                                                                                                    |
| ...    | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                                                                                                                                                       |

### Value

data.frame with 5 columns: region\_code, channelId, title, etag, id

### References

<https://developers.google.com/youtube/v3/docs/guideCategories/list>

### Examples

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
list_guidecats(c(region_code = "JP"))  
  
## End(Not run)
```

---

|            |                                                       |
|------------|-------------------------------------------------------|
| list_langs | <i>List Languages That YouTube Currently Supports</i> |
|------------|-------------------------------------------------------|

---

**Description**

List Languages That YouTube Currently Supports

**Usage**

```
list_langs(hl = NULL, ...)
```

**Arguments**

|     |                                                                                                                               |
|-----|-------------------------------------------------------------------------------------------------------------------------------|
| hl  | Language used for text values. Optional. Default is en-US. For other allowed language codes, see <a href="#">list_langs</a> . |
| ... | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                    |

**Value**

data.frame with 3 columns: hl (two letter abbreviation), name (of the language), etag

**References**

<https://developers.google.com/youtube/v3/docs/i18nLanguages/list>

**Examples**

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
list_langs()  
  
## End(Not run)
```

---

|                |                       |
|----------------|-----------------------|
| list_my_videos | <i>List My videos</i> |
|----------------|-----------------------|

---

**Description**

List My videos

**Usage**

```
list_my_videos(...)
```

**Arguments**

... additional arguments to pass to [list\\_channel\\_videos](#)

**Value**

data.frame with each list corresponding to a different playlist

**Examples**

```
## Not run:
  list_my_videos()

## End(Not run)
```

---

|              |                                                             |
|--------------|-------------------------------------------------------------|
| list_regions | <i>List Content Regions That YouTube Currently Supports</i> |
|--------------|-------------------------------------------------------------|

---

**Description**

List Content Regions That YouTube Currently Supports

**Usage**

```
list_regions(hl = NULL, ...)
```

**Arguments**

hl Language used for text values. Optional. Default is en-US. For other allowed language codes, see [list\\_langs](#).

... Additional arguments passed to [tuber\\_GET](#).

**Value**

data.frame with 3 columns: gl (two letter abbreviation), name (of the region), etag

**References**

<https://developers.google.com/youtube/v3/docs/i18nRegions/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

list_regions()

## End(Not run)
```

---

|                |                                                              |
|----------------|--------------------------------------------------------------|
| list_videocats | <i>List of Categories That Can be Associated with Videos</i> |
|----------------|--------------------------------------------------------------|

---

## Description

List of Categories That Can be Associated with Videos

## Usage

```
list_videocats(filter = NULL, ...)
```

## Arguments

|        |                                                                                                                                                                                                                                                                                        |
|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| filter | string; Required. named vector of length 1 potential names of the entry in the vector: region_code: Character. Required. Has to be a ISO 3166-1 alpha-2 code (see <a href="https://www.iso.org/obp/ui/#search">https://www.iso.org/obp/ui/#search</a> ) category_id: video category ID |
| ...    | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                                                                                                                                             |

## Value

data.frame with 6 columns: region\_code, channelId, title, assignable, etag, id

## References

<https://developers.google.com/youtube/v3/docs/videoCategories/list>

## Examples

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
list_videocats(c(region_code = "JP"))  
list_videocats() # Will throw an error asking for a valid filter with valid region_code  
  
## End(Not run)
```

---

|             |                            |
|-------------|----------------------------|
| list_videos | List (Most Popular) Videos |
|-------------|----------------------------|

---

**Description**

List (Most Popular) Videos

**Usage**

```
list_videos(  
  part = "contentDetails",  
  max_results = 50,  
  page_token = NULL,  
  hl = NULL,  
  region_code = NULL,  
  video_category_id = NULL,  
  ...  
)
```

**Arguments**

|                   |                                                                                                                                                                                                                                                                           |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| part              | Required. Comma separated string including one or more of the following: contentDetails,fileDetails, id, liveStreamingDetails, localizations, player,processingDetails, recordingDetails,snippet, statistics, status, suggestions, topicDetails. Default: contentDetails. |
| max_results       | Maximum number of items that should be returned. Integer. Optional. Default is 50. Values over 50 will trigger multiple requests and may use additional API quota.                                                                                                        |
| page_token        | specific page in the result set that should be returned, optional                                                                                                                                                                                                         |
| hl                | Language used for text values. Optional. Default is en-US. For other allowed language codes, see <a href="#">list_langs</a> .                                                                                                                                             |
| region_code       | Character. Required. Has to be a ISO 3166-1 alpha-2 code (see <a href="https://www.iso.org/obp/ui/#search">https://www.iso.org/obp/ui/#search</a> ).                                                                                                                      |
| video_category_id | the video category for which the chart should be retrieved. See also <a href="#">list_videocats</a> .                                                                                                                                                                     |
| ...               | Additional arguments passed to <a href="#">tuber_GET</a> .                                                                                                                                                                                                                |

**Value**

data.frame with 5 columns: channelId, title, assignable, etag, id

**References**

<https://developers.google.com/youtube/v3/docs/search/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

list_videos()

## End(Not run)
```

---

|                  |                                     |
|------------------|-------------------------------------|
| quota_management | <i>YouTube API Quota Management</i> |
|------------------|-------------------------------------|

---

**Description**

Functions to track and manage YouTube API quota usage

---

|          |                      |
|----------|----------------------|
| read_sbv | <i>Read SBV file</i> |
|----------|----------------------|

---

**Description**

Read SBV file

**Usage**

```
read_sbv(file)
```

**Arguments**

|      |                               |
|------|-------------------------------|
| file | The file name of the sbv file |
|------|-------------------------------|

**Value**

A data.frame with start/stop times and the text

**Examples**

```
if (yt_authorized()){
  vids <- list_my_videos()
  res <- list_caption_tracks(video_id = vids$contentDetails.videoId[1])
  cap <- get_captions(id = res$id, as_raw = FALSE)
  tfile <- tempfile(fileext = ".sbv")
  writeLines(cap, tfile)
  x <- read_sbv(tfile)
  if (requireNamespace("hms", quietly = TRUE)) {
    x$start <- hms::as_hms(x$start)
    x$stop <- hms::as_hms(x$stop)
  }
}
```

---

|       |                                                     |
|-------|-----------------------------------------------------|
| tuber | <b>tuber</b> provides access to the YouTube API V3. |
|-------|-----------------------------------------------------|

---

**Description**

**tuber** provides access to the YouTube API V3 via RESTful calls.

---

|             |                                      |
|-------------|--------------------------------------|
| tuber_check | <i>Request Response Verification</i> |
|-------------|--------------------------------------|

---

**Description**

Request Response Verification

**Usage**

tuber\_check(req)

**Arguments**

req                      request

**Value**

in case of failure, a message

---

|              |               |
|--------------|---------------|
| tuber_DELETE | <i>DELETE</i> |
|--------------|---------------|

---

**Description**

DELETE

**Usage**

tuber\_DELETE(path, query, ...)

**Arguments**

path                      path to specific API request URL  
query                      query list  
...                          Additional arguments passed to [GET](#).

**Value**

list

---

`tuber_GET`*GET*

---

**Description**

GET

**Usage**`tuber_GET(path, query, auth = "token", ...)`**Arguments**

|                    |                                                                                        |
|--------------------|----------------------------------------------------------------------------------------|
| <code>path</code>  | path to specific API request URL                                                       |
| <code>query</code> | query list                                                                             |
| <code>auth</code>  | A character vector of the authentication method, either "token" (the default) or "key" |
| <code>...</code>   | Additional arguments passed to <a href="#">GET</a> .                                   |

**Value**list

---

`tuber_POST`*POST*

---

**Description**

POST

**Usage**`tuber_POST(path, query, body = "", ...)`**Arguments**

|                    |                                                      |
|--------------------|------------------------------------------------------|
| <code>path</code>  | path to specific API request URL                     |
| <code>query</code> | query list                                           |
| <code>body</code>  | passing image through body                           |
| <code>...</code>   | Additional arguments passed to <a href="#">GET</a> . |

**Value**

list

---

|                 |                             |
|-----------------|-----------------------------|
| tuber_POST_json | <i>POST encoded in json</i> |
|-----------------|-----------------------------|

---

**Description**

POST encoded in json

**Usage**

```
tuber_POST_json(path, query, body = "", ...)
```

**Arguments**

|       |                                                      |
|-------|------------------------------------------------------|
| path  | path to specific API request URL                     |
| query | query list                                           |
| body  | passing image through body                           |
| ...   | Additional arguments passed to <a href="#">GET</a> . |

**Value**

list

---

|               |                                              |
|---------------|----------------------------------------------|
| unicode_utils | <i>Unicode and Text Processing Utilities</i> |
|---------------|----------------------------------------------|

---

**Description**

Internal functions for consistent text and Unicode handling across tuber

---

|                       |                                          |
|-----------------------|------------------------------------------|
| update_video_metadata | <i>Update a YouTube Video's Metadata</i> |
|-----------------------|------------------------------------------|

---

**Description**

This function updates the metadata of an existing YouTube video using the YouTube Data API.

**Usage**

```
update_video_metadata(  
  video_id,  
  title,  
  category_id,  
  description,  
  privacy_status,  
  made_for_kids  
)
```

**Arguments**

- video\_id      A character string specifying the ID of the video you want to update.
- title         A character string specifying the new title for the video.
- category\_id   A character string specifying the new category ID for the video.
- description   A character string specifying the new description for the video.
- privacy\_status A character string specifying the new privacy status for the video ('public', 'private', or 'unlisted').
- made\_for\_kids A boolean specifying whether the video is self-declared as made for kids.

**Value**

A list containing the server response after the update attempt.

**Examples**

```
## Not run:
update_video_metadata(video_id = "YourVideoID",
                      title = "New Video Title",
                      category_id = "24",
                      description = "New Description",
                      privacy_status = "public",
                      made_for_kids = FALSE)

## End(Not run)
```

---

|                |                                        |
|----------------|----------------------------------------|
| upload_caption | <i>Upload Video Caption to Youtube</i> |
|----------------|----------------------------------------|

---

**Description**

Upload Video Caption to Youtube

**Usage**

```
upload_caption(
  file,
  video_id,
  language = "en-US",
  caption_name,
  is_draft = FALSE,
  query = NULL,
  open_url = FALSE,
  ...
)
```

**Arguments**

|              |                                                                                                                                                                                   |
|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| file         | Filename of the caption, probably ‘.srt’                                                                                                                                          |
| video_id     | YouTube Video ID. Try <a href="#">list_my_videos</a> for examples.                                                                                                                |
| language     | character string of ‘BCP47’ language type. See <a href="https://www.rfc-editor.org/rfc/bcp/bcp47.txt">https://www.rfc-editor.org/rfc/bcp/bcp47.txt</a> for language specification |
| caption_name | character vector of the name for the caption.                                                                                                                                     |
| is_draft     | logical indicating whether the caption track is a draft.                                                                                                                          |
| query        | Fields for ‘query’ in ‘POST’                                                                                                                                                      |
| open_url     | Should the video be opened using <a href="#">browseURL</a>                                                                                                                        |
| ...          | Additional arguments to send to <a href="#">POST</a>                                                                                                                              |

**Value**

A list of the response object from the [POST](#), content, and the URL of the video

**Note**

See <https://developers.google.com/youtube/v3/docs/captions#resource> for full specification

**Examples**

```
## Not run:
xx = list_my_videos()
video_id = xx$contentDetails.videoId[1]
video_id = as.character(video_id)
language = "en-US"

## End(Not run)
```

---

|              |                                |
|--------------|--------------------------------|
| upload_video | <i>Upload Video to Youtube</i> |
|--------------|--------------------------------|

---

**Description**

Upload Video to Youtube

**Usage**

```
upload_video(
  file,
  snippet = NULL,
  status = list(privacyStatus = "public"),
  query = NULL,
  open_url = FALSE,
  ...
)
```

**Arguments**

|          |                                                                                                                                                                                                                                                                |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| file     | Filename of the video locally                                                                                                                                                                                                                                  |
| snippet  | Additional fields for the video, including 'description' and 'title'. See <a href="https://developers.google.com/youtube/v3/docs/videos#resource">https://developers.google.com/youtube/v3/docs/videos#resource</a> for other fields. Coerced to a JSON object |
| status   | Additional fields to be put into the status input. options for 'status' are 'license' (which should hold: 'creativeCommon', or 'youtube'), 'privacyStatus', 'publicStatsViewable', 'publishAt'.                                                                |
| query    | Fields for 'query' in 'POST'                                                                                                                                                                                                                                   |
| open_url | Should the video be opened using <a href="#">browseURL</a>                                                                                                                                                                                                     |
| ...      | Additional arguments to send to <a href="#">tuber_POST</a> and therefore <a href="#">POST</a>                                                                                                                                                                  |

**Value**

A list of the response object from the [POST](#), content, and the URL of the uploaded

**Note**

The information for 'status' and 'snippet' are at <https://developers.google.com/youtube/v3/docs/videos#resource> but the subset of these fields to pass in are located at: <https://developers.google.com/youtube/v3/docs/videos/insert> The 'part' parameter serves two purposes in this operation. It identifies the properties that the write operation will set, this will be automatically detected by the names of 'body'. See <https://developers.google.com/youtube/v3/docs/videos/insert#usage>

**Examples**

```
snippet = list(
  title = "Test Video",
  description = "This is just a random test.",
  tags = c("r language", "r programming", "data analysis")
)
status = list(privacyStatus = "private")
```

---

|                    |                                |
|--------------------|--------------------------------|
| yt_get_quota_usage | <i>Get Current Quota Usage</i> |
|--------------------|--------------------------------|

---

**Description**

Returns the current estimated quota usage for the day

**Usage**

```
yt_get_quota_usage()
```

**Value**

List with quota\_used, quota\_limit, quota\_remaining, and reset\_time

**Examples**

```
## Not run:
quota_status <- yt_get_quota_usage()
cat("Used:", quota_status$quota_used, "/", quota_status$quota_limit)

## End(Not run)
```

---

|        |                               |
|--------|-------------------------------|
| yt_key | <i>Manage YouTube API key</i> |
|--------|-------------------------------|

---

**Description**

These functions manage your YouTube API key and package key in .Renviron.

**Usage**

```
yt_get_key(decrypt = FALSE)
yt_set_key(key, type)
```

**Arguments**

|         |                                                                                                                                                                                                                                                     |
|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| decrypt | A boolean vector specifying whether to decrypt the supplied key with 'httr2::secret_decrypt()'. Defaults to 'FALSE'. If 'TRUE', requires the environment variable 'TUBER_KEY' to be set in '.Renviron'.                                             |
| key     | A character vector specifying a YouTube API key.                                                                                                                                                                                                    |
| type    | A character vector specifying the type of API key to set. One of 'api' (the default, stored in 'YOUTUBE_KEY') or 'package'. Package keys are stored in 'TUBER_KEY' and are used to decrypt API keys, for use in continuous integration and testing. |

**Value**

'yt\_get\_key()' returns a character vector with the YouTube API key stored in '.Renviron'. If this value is not stored in '.Renviron', the functions return 'NULL'.

When the 'type' argument is set to 'api', 'yt\_set\_key()' assigns a YouTube API key to 'YOUTUBE\_KEY' in '.Renviron' and invisibly returns 'NULL'. When the 'type' argument is set to 'package', 'yt\_set\_key()' assigns a package key to 'TUBER\_KEY' in '.Renviron' and invisibly returns 'NULL'.

**Examples**

```
## Not run:
## for interactive use
yt_get_key()

list_channel_videos(
  channel_id = "UCDgj5-mFohWZ5irWSFMFcng",
  max_results = 3,
  part = "snippet",
  auth = "key"
)

## for continuous integration and testing
yt_set_key(httr2::secret_make_key(), type = "package")
x <- httr2::secret_encrypt("YOUR_YOUTUBE_API_KEY", "TUBER_KEY")
yt_set_key(x, type = "api")
yt_get_key(decrypt = TRUE)

list_channel_videos(
  channel_id = "UCDgj5-mFohWZ5irWSFMFcng",
  max_results = 3,
  part = "snippet",
  auth = "key"
)

## End(Not run)
```

yt\_oauth

*Set up Authorization***Description**

The function looks for `.httr-oauth` in the working directory. If it doesn't find it, it expects an application ID and a secret. If you want to remove the existing `.httr-oauth`, set `remove_old_oauth` to `TRUE`. By default, it is set to `FALSE`. The function launches a browser to allow you to authorize the application

**Usage**

```
yt_oauth(
  app_id = NULL,
  app_secret = NULL,
  scope = "ssl",
  token = ".httr-oauth",
  ...
)
```

**Arguments**

|            |                                                                                                                                                                                                                             |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| app_id     | client id; required; no default                                                                                                                                                                                             |
| app_secret | client secret; required; no default                                                                                                                                                                                         |
| scope      | Character. ssl, basic, own_account_readonly, upload_and_manage_own_videos, partner, and partner_audit. Required. ssl and basic are basically interchangeable. Default is ssl.                                               |
| token      | path to file containing the token. If a path is given, the function will first try to read from it. Default is .httr-oauth in the local directory. So if there is such a file, the function will first try to read from it. |
| ...        | Additional arguments passed to <a href="#">oauth2.0_token</a>                                                                                                                                                               |

**Details**

If a browser cannot be opened, pass `use_oob = TRUE` to `yt_oauth()` so authentication can be completed using an out-of-band code. Delete the `.httr-oauth` file in the working directory to force re-authentication.

**Value**

sets the `google_token` option and also saves `.httr_oauth` in the working directory (find out the working directory via `getwd()`)

**References**

<https://developers.google.com/youtube/v3/docs/>

<https://developers.google.com/youtube/v3/guides/auth/client-side-web-apps> for different scopes

**Examples**

```
## Not run:
yt_oauth(paste0("998136489867-5t3tq1g7hbovoj46dreqd6k5kd35ctjn",
               ".apps.googleusercontent.com"),
         "Mb0St6cQhhFkwETXKur-L9rN")

## End(Not run)
```

---

yt\_reset\_quota

*Reset Quota Counter*


---

**Description**

Reset the quota counter (typically done automatically at midnight UTC)

**Usage**

```
yt_reset_quota()
```

yt\_search

*Search YouTube***Description**

Search for videos, channels and playlists. (By default, the function searches for videos.)

**Usage**

```
yt_search(
  term = NULL,
  max_results = 50,
  channel_id = NULL,
  channel_type = NULL,
  type = "video",
  event_type = NULL,
  location = NULL,
  location_radius = NULL,
  published_after = NULL,
  published_before = NULL,
  video_definition = "any",
  video_caption = "any",
  video_license = "any",
  video_syndicated = "any",
  region_code = NULL,
  relevance_language = "en",
  video_type = "any",
  simplify = TRUE,
  get_all = TRUE,
  page_token = NULL,
  max_pages = Inf,
  ...
)
```

**Arguments**

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| term        | Character. Search term; required; no default For using Boolean operators, see the API documentation. Here's some of the relevant information: "Your request can also use the Boolean NOT (-) and OR ( ) operators to exclude videos or to find videos that are associated with one of several search terms. For example, to search for videos matching either "boating" or "sailing", set the q parameter value to boating sailing. Similarly, to search for videos matching either "boating" or "sailing" but not "fishing", set the q parameter value to boating sailing-fishing" |
| max_results | Maximum number of items that should be returned in total. Integer. Optional. Can be between 1 and 500. Default is 50. If get_all = TRUE, multiple API calls are made until this many results are collected (subject to YouTube limits).                                                                                                                                                                                                                                                                                                                                             |

|                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                    | Requesting a large number of results will consume more API quota. Search results are constrained to a maximum of 500 videos if type is video and we have a value of channel_id.                                                                                                                                                                                                                                                                           |
| channel_id         | Character. Only return search results from this channel; Optional.                                                                                                                                                                                                                                                                                                                                                                                        |
| channel_type       | Character. Optional. Takes one of two values: 'any', 'show'. Default is 'any'                                                                                                                                                                                                                                                                                                                                                                             |
| type               | Character. Optional. Takes one of three values: 'video', 'channel', 'playlist'. Default is 'video'.                                                                                                                                                                                                                                                                                                                                                       |
| event_type         | Character. Optional. Takes one of three values: 'completed', 'live', 'upcoming'                                                                                                                                                                                                                                                                                                                                                                           |
| location           | Character. Optional. Latitude and Longitude within parentheses, e.g. "(37.42307,-122.08427)"                                                                                                                                                                                                                                                                                                                                                              |
| location_radius    | Character. Optional. e.g. "1500m", "5km", "10000ft", "0.75mi"                                                                                                                                                                                                                                                                                                                                                                                             |
| published_after    | Character. Optional. RFC 339 Format. For instance, "1970-01-01T00:00:00Z"                                                                                                                                                                                                                                                                                                                                                                                 |
| published_before   | Character. Optional. RFC 339 Format. For instance, "1970-01-01T00:00:00Z"                                                                                                                                                                                                                                                                                                                                                                                 |
| video_definition   | Character. Optional. Takes one of three values: 'any' (return all videos; Default), 'high', 'standard'                                                                                                                                                                                                                                                                                                                                                    |
| video_caption      | Character. Optional. Takes one of three values: 'any' (return all videos; Default), 'closedCaption', 'none'. Type must be set to video.                                                                                                                                                                                                                                                                                                                   |
| video_license      | Character. Optional. Takes one of three values: 'any' (return all videos; Default), 'creativeCommon' (return videos with Creative Commons license), 'youtube' (return videos with standard YouTube license).                                                                                                                                                                                                                                              |
| video_syndicated   | Character. Optional. Takes one of two values: 'any' (return all videos; Default), 'true' (return only syndicated videos)                                                                                                                                                                                                                                                                                                                                  |
| region_code        | Character. Required. Has to be a ISO 3166-1 alpha-2 code (see <a href="https://www.iso.org/obp/ui/#search">https://www.iso.org/obp/ui/#search</a> ).                                                                                                                                                                                                                                                                                                      |
| relevance_language | Character. Optional. The relevance_language argument instructs the API to return search results that are most relevant to the specified language. The parameter value is typically an ISO 639-1 two-letter language code. However, you should use the values zh-Hans for simplified Chinese and zh-Hant for traditional Chinese. Please note that results in other languages will still be returned if they are highly relevant to the search query term. |
| video_type         | Character. Optional. Takes one of three values: 'any' (return all videos; Default), 'episode' (return episode of shows), 'movie' (return movies)                                                                                                                                                                                                                                                                                                          |
| simplify           | Boolean. Return a data.frame if TRUE. Default is TRUE. If TRUE, it returns a list that carries additional information.                                                                                                                                                                                                                                                                                                                                    |
| get_all            | get all results, iterating through all the results pages. Default is TRUE. Result is a data.frame. Optional.                                                                                                                                                                                                                                                                                                                                              |
| page_token         | specific page in the result set that should be returned, optional                                                                                                                                                                                                                                                                                                                                                                                         |
| max_pages          | Maximum number of pages to retrieve when get_all is TRUE. Default is Inf (no page limit). Setting a lower value can reduce API quota usage.                                                                                                                                                                                                                                                                                                               |
| ...                | Additional arguments passed to <code>tuber_GET</code> .                                                                                                                                                                                                                                                                                                                                                                                                   |

**Value**

data.frame with 16 elements: video\_id, publishedAt, channelId, title, description, thumbnails.default.url, thumbnails.default.width, thumbnails.default.height, thumbnails.medium.url, thumbnails.medium.width, thumbnails.medium.height, thumbnails.high.url, thumbnails.high.width, thumbnails.high.height, channelTitle, liveBroadcastContent The returned data.frame also has the following attributes:  
 total\_results: The total number of results reported by the API  
 actual\_results: The actual number of rows returned  
 api\_limit\_reached: Whether the YouTube API result limit was reached

**References**

<https://developers.google.com/youtube/v3/docs/search/list>

**Examples**

```
## Not run:

# Set API token via yt_oauth() first

yt_search(term = "Barack Obama")
yt_search(term = "Barack Obama", published_after = "2016-10-01T00:00:00Z")
yt_search(term = "Barack Obama", published_before = "2016-09-01T00:00:00Z")
yt_search(term = "Barack Obama", published_before = "2016-03-01T00:00:00Z",
          published_after = "2016-02-01T00:00:00Z")
yt_search(term = "Barack Obama", published_before = "2016-02-10T00:00:00Z",
          published_after = "2016-01-01T00:00:00Z")

# To check how many results were found vs. how many were returned:
results <- yt_search(term = "drone videos")
attr(results, "total_results") # Total number reported by YouTube
attr(results, "actual_results") # Number actually returned
attr(results, "api_limit_reached") # Whether API limit was reached

## End(Not run)
```

---

|                    |                        |
|--------------------|------------------------|
| yt_set_quota_limit | <i>Set Quota Limit</i> |
|--------------------|------------------------|

---

**Description**

Set the daily quota limit (default is 10,000 units)

**Usage**

```
yt_set_quota_limit(limit)
```

**Arguments**

|       |                                     |
|-------|-------------------------------------|
| limit | Integer. Daily quota limit in units |
|-------|-------------------------------------|

Examples

```
## Not run:
# If you have a higher quota limit
yt_set_quota_limit(50000)

## End(Not run)
```

---

|          |                                                    |
|----------|----------------------------------------------------|
| yt_token | <i>Check if authentication token is in options</i> |
|----------|----------------------------------------------------|

---

Description

Check if authentication token is in options

Usage

```
yt_token()

yt_authorized()

yt_check_token()
```

Value

A Token2.0 class

---

|                 |                                                                    |
|-----------------|--------------------------------------------------------------------|
| yt_topic_search | <i>Search YouTube by Topic It uses the Freebase list of topics</i> |
|-----------------|--------------------------------------------------------------------|

---

Description

Search YouTube by Topic It uses the Freebase list of topics

Usage

```
yt_topic_search(topic = NULL, ...)
```

Arguments

- topic            topic being searched for; required; no default
- ...            Additional arguments passed to [tuber\\_GET](#).

Value

a list

**Examples**

```
## Not run:  
  
# Set API token via yt_oauth() first  
  
yt_topic_search(topic = "Barack Obama")  
  
## End(Not run)
```

# Index

add\_video\_to\_playlist, 3

browseURL, 42, 43

change\_playlist\_title, 4

create\_playlist, 4

delete\_captions, 5

delete\_channel\_sections, 6

delete\_comments, 6

delete\_playlist\_items, 8

delete\_playlists, 7

delete\_videos, 8

GET, 25, 38–40

get\_all\_channel\_video\_stats, 9

get\_all\_comments, 10

get\_captions, 11

get\_channel\_stats, 12

get\_comment\_threads, 14

get\_comments, 13

get\_playlist\_item\_ids, 18

get\_playlist\_item\_videoids, 19

get\_playlist\_items, 16

get\_playlists, 15

get\_related\_videos, 20

get\_stats, 21

get\_subscriptions, 22

get\_video\_details, 23

list\_abuse\_report\_reasons, 24

list\_caption\_tracks, 11, 26

list\_captions, 25

list\_channel\_activities, 27

list\_channel\_resources, 28

list\_channel\_sections, 30

list\_channel\_videos, 31, 34

list\_guidecats, 32

list\_langs, 16, 24, 29–33, 33, 34, 36

list\_my\_channel (get\_channel\_stats), 12

list\_my\_videos, 33, 42

list\_regions, 27, 34

list\_videocats, 35, 36

list\_videos, 36

oauth2.0\_token, 46

POST, 42, 43

quota\_management, 37

read\_sbv, 37

tuber, 38

tuber\_check, 38

tuber\_DELETE, 5–8, 38

tuber\_GET, 9–13, 15–27, 29–36, 39, 48, 50

tuber\_POST, 4, 39, 43

tuber\_POST\_json, 3, 40

unicode\_utils, 40

update\_video\_metadata, 40

upload\_caption, 41

upload\_video, 42

yt\_authorized (yt\_token), 50

yt\_check\_token (yt\_token), 50

yt\_get\_key (yt\_key), 44

yt\_get\_quota\_usage, 43

yt\_key, 44

yt\_oauth, 45

yt\_reset\_quota, 46

yt\_search, 47

yt\_set\_key (yt\_key), 44

yt\_set\_quota\_limit, 49

yt\_token, 50

yt\_topic\_search, 50