

# Package: mediacloudr (via r-universe)

September 1, 2024

**Type** Package

**Title** Wrapper for the 'mediacloud.org' API

**Version** 0.1.1.9000

**Depends** R (>= 3.2.0)

**Description** API wrapper to gather news stories, media information and tags from the 'mediacloud.org' API, based on a multilevel query <<https://mediacloud.org/>>. A personal API key is required.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Imports** httr, jsonlite, rvest, xml2

**Suggests** testthat, covr, knitr, rmarkdown

**RoxygenNote** 6.1.1

**VignetteBuilder** knitr

**Repository** <https://jandix.r-universe.dev>

**RemoteUrl** <https://github.com/jandix/mediacloudr>

**RemoteRef** HEAD

**RemoteSha** c1d997690e820ac957cdcc4421e9b0e972caea42

## Contents

|                             |   |
|-----------------------------|---|
| extract_meta_data . . . . . | 2 |
| get_media_source . . . . .  | 2 |
| get_story . . . . .         | 3 |
| get_story_list . . . . .    | 4 |
| meta_data_html . . . . .    | 5 |

|              |          |
|--------------|----------|
| <b>Index</b> | <b>6</b> |
|--------------|----------|

---

extract\_meta\_data      *Extract meta data*

---

### Description

extract\_meta\_data extracts native, open graph and twitter meta data from html documents. The meta data include url, title, description and image. The html document is parsed within the function

### Usage

```
extract_meta_data(html_doc)
```

### Arguments

html\_doc      Character string including the html document.

### Value

List with three sublists for native, open graph and twitter.

### Examples

```
## Not run:
library(httr)
url <- "https://bits.blogs.nytimes.com/2013/04/07/the-potential-and-the-risks-of-data-science"
response <- GET(url)
html_document <- content(response, type = "text", encoding = "UTF-8")
meta_data <- extract_meta_data(html_doc = html_document)

## End(Not run)
```

---

get\_media\_source      *Get media by id*

---

### Description

get\_media returns media source by their id. A media source is one publisher. Every story that can be collected via get\_story or get\_story\_list belongs to one media source.

### Usage

```
get_media_source(media_id, api_key = Sys.getenv("MEDIACLOUD_API_KEY"))
```

**Arguments**

|          |   |
|----------|---|
| media_id | Positive integer that contains a valid media“ id.   |
| api_key  | Character string with the API key you get from mediacloud.org. Passing it is compulsory. Alternatively, function can be provided from the global environment. |

**Value**

Data frame with results. See [https://github.com/berkmancenter/mediacloud/blob/master/doc/api\\_2\\_0\\_spec/api\\_2\\_0\\_spec.md#media](https://github.com/berkmancenter/mediacloud/blob/master/doc/api_2_0_spec/api_2_0_spec.md#media) for field descriptions.

**Examples**

```
## Not run:
media_source <- get_media_source(media_id = 604L)

## End(Not run)
```

---

|           |                        |
|-----------|------------------------|
| get_story | <i>Get story by id</i> |
|-----------|------------------------|

---

**Description**

get\_story returns news stories by their id. One story represents one online publication. Each story refers to a single URL from any feed within a single media source.

**Usage**

```
get_story(story_id, api_key = Sys.getenv("MEDIACLOUD_API_KEY"))
```

**Arguments**

|          |   |
|----------|---|
| story_id | Positive numeric that contains a valid story id.  |
| api_key  | Character string with the API key you get from mediacloud.org. Passing it is compulsory. Alternatively, function can be provided from the global environment. |

**Value**

Data frame with results. See [https://github.com/berkmancenter/mediacloud/blob/master/doc/api\\_2\\_0\\_spec/api\\_2\\_0\\_spec.md#stories](https://github.com/berkmancenter/mediacloud/blob/master/doc/api_2_0_spec/api_2_0_spec.md#stories) for field descriptions.

**Examples**

```
## Not run:
story <- get_story(story_id = 604L)

## End(Not run)
```

---

|                |                       |
|----------------|-----------------------|
| get_story_list | <i>Get story list</i> |
|----------------|-----------------------|

---

**Description**

get\_story returns a list of stories based on a multifaceted query. One story represents one online publication. Each story refers to a single URL from any feed within a single media source.

**Usage**

```
get_story_list(last_process_stories_id = 0L, rows = 100,
  feeds_id = NULL, q = NULL, fq = NULL,
  sort = "processed_stories_id", wc = FALSE, show_feeds = FALSE,
  api_key = Sys.getenv("MEDIACLOUD_API_KEY"))
```

**Arguments**

|                         |   |
|-------------------------|---|
| last_process_stories_id | Return stories in which the processed_stories_id is greater than this value.  |
| rows                    | Number of stories to return, max 1000.  |
| feeds_id                | Return only stories that match the given feeds_id, sorted by descending publish date  |
| q                       | If specified, return only results that match the given Solr query. Only one q parameter may be included.  |
| fq                      | If specified, filter results by the given Solr query. More than one fq parameter may be included.   |
| sort                    | Returned results sort order. Supported values: processed_stories_id, random   |
| wc                      | If set to TRUE, include a 'word_count' field with each story that includes a count of the most common words in the story                                      |
| show_feeds              | If set to TRUE, include a 'feeds' field with a list of the feeds associated with this story   |
| api_key                 | Character string with the API key you get from mediacloud.org. Passing it is compulsory. Alternatively, function can be provided from the global environment. |

**Value**

Data frame with results. See [https://github.com/berkmancenter/mediacloud/blob/master/doc/api\\_2\\_0\\_spec/api\\_2\\_0\\_spec.md#stories](https://github.com/berkmancenter/mediacloud/blob/master/doc/api_2_0_spec/api_2_0_spec.md#stories) for field descriptions.

### Examples

```
## Not run:  
stories <- get_story_list()  
stories <- get_story_list(q = "Trump")  
  
## End(Not run)
```

---

|                |  |
|----------------|--|
| meta_data_html | <i>HTML document to test extract_meta_data</i> |
|----------------|--|

---

### Description

A HTML document with basic meta tags for open-graph, twitter and native meta data.

### Usage

```
meta_data_html
```

### Format

An object of class character of length 1.

# Index

## \* datasets

meta\_data\_html, [5](#)

extract\_meta\_data, [2](#)

get\_media\_source, [2](#)

get\_story, [3](#)

get\_story\_list, [4](#)

meta\_data\_html, [5](#)