# Package: ralger (via r-universe)

October 10, 2026

**Type** Package

**Title** Easy Web Scraping

**Version** 2.3.0

**Maintainer** Mohamed El Fodil Ihaddaden <ihaddaden.fodeil@gmail.com>

**Description** The goal of 'ralger' is to facilitate web scraping in R.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**URL** https://github.com/feddelegrand7/ralger

**BugReports** https://github.com/feddelegrand7/ralger/issues

**VignetteBuilder** knitr

**Imports** rvest, xml2, tidyr, dplyr, stringr, robotstxt, crayon, curl, stringi, urltools (>= 1.7.3), purrr (>= 1.0.2)

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

**RoxygenNote** 7.3.1

**Config/pak/sysreqs** libicu-dev libxml2-dev libssl-dev

**Repository** https://feddelegrand7.r-universe.dev

**Date/Publication** 2026-03-14 12:18:58 UTC

**RemoteUrl** https://github.com/feddelegrand7/ralger

**RemoteRef** HEAD

**RemoteSha** 84d9a2a15ddf43660f81859f76f467324b898ee4

## Contents


attribute_scrap . . . . . 2
audio_scrap . . . . . 3
colors_scrap . . . . . 4
comments_scrap . . . . . 4
csv_scrap . . . . . 5

images_noalt_scrap . . . . . 6
images_preview . . . . . 6
images_scrap . . . . . 7
paragraphs_scrap . . . . . 8
pdf_scrap . . . . . 9
scrap . . . . . 9
table_scrap . . . . . 10
tidy_scrap . . . . . 11
titles_scrap . . . . . 11
weblink_scrap . . . . . 12
xls_scrap . . . . . 13
xlsx_scrap . . . . . 13

**Index** **15**

---

`attribute_scrap` *Scraping attributes from HTML elements*

---

## Description

This function is used to scrape attributes from HTML elements

## Usage

```
attribute_scrap(link, node, attr, askRobot = FALSE)
```

## Arguments

| | |
|---|---|
| `link` | the link of the web page to scrape |
| `node` | the HTML element to consider |
| `attr` | the attribute to scrape |
| `askRobot` | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

## Value

a character vector.

## Examples

```
# Extracting the web links within the World Bank research and publications page

link <- "https://ropensci.org/"

# scraping the class attributes' names from all the anchor

attribute_scrap(link = link, node = "a", attr = "class")
```

audio_scrap *Scrape Audio Files from a Web Page*

## Description

This function is used to scrape audio file URLs from a web page and optionally download them. It searches both <audio> tags and <a> (anchor) tags for links matching the specified audio extensions.

## Usage

```
audio_scrap(link, extensions = c("mp3", "wav"), path = getwd(), askRobot = FALSE)
```

## Arguments

| | |
|---|---|
| link | the link of the web page to scrape |
| extensions | a character vector of audio file extensions to filter by (without the leading dot). Defaults to c("mp3", "wav"). |
| path | the path where audio files will be downloaded. Defaults to the current working directory. Set to NULL to return URLs without downloading. |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page? Default is FALSE. |

## Value

called for the side effect of downloading audio files. Returns the vector of matched audio URLs invisibly, or NULL if none are found.

## Examples

```
## Not run:

# Scrape and download mp3 and wav files from a page
audio_scrap(
  link = "https://www.example.com/podcasts",
  extensions = c("mp3", "wav"),
  path = getwd()
)

# Return audio URLs without downloading
audio_scrap(
  link = "https://www.example.com/podcasts",
  extensions = "mp3",
  path = NULL
)


## End(Not run)
```

## colors_scrap *Scrape Color Codes from a Web Page*

### Description

This function scrapes all color codes present within a given website. It searches inline `style` attributes, `<style>` tags, and linked external CSS stylesheets for color values in the following formats: hexadecimal (`#RGB`, `#RRGGBB`, `#RRGGBBAA`), `rgb()`, `rgba()`, `hsl()`, and `hsla()`.

### Usage

```
colors_scrap(link, askRobot = FALSE)
```

### Arguments

| | |
|---|---|
| `link` | the link of the web page to scrape. Can be a character vector of multiple URLs. |
| `askRobot` | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page? Default is `FALSE`. |

### Value

a character vector of unique color codes found on the page, or `invisible(NULL)` if none are found, or `NA` on error.

### Examples

```
colors_scrap(link = "https://ropensci.org/")
```

## comments_scrap *Scrape HTML comments from a web page*

### Description

Extracts HTML comments (<!– comment –>) from a webpage. Useful for detecting hidden notes, debug info, or developer messages.

### Usage

```
comments_scrap(link, askRobot = FALSE)
```

### Arguments

| | |
|---|---|
| link | Character. The URL of the web page to scrape. |
| askRobot | Logical. Should the function check robots.txt before scraping? Default is FALSE. |

### Value

A character vector of HTML comments found on the page.

### Examples

```
link <- "https://example.com"
comments_scrap(link)
```

---

## csv_scrap *Scrape and download CSV files from a Web Page*

---

### Description

Scrape and download CSV files from a Web Page

### Usage

```
csv_scrap(link, path = getwd(), askRobot = FALSE)
```

### Arguments

| | |
|---|---|
| link | the link of the web page |
| path | the path where to save the CSV files. Defaults to the current directory |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

### Value

called for the side effect of downloading CSV files from a website

## images_noalt_scrap *Scrape Images URLS that don't have 'alt' attributes*

### Description

Scrape Images URLS that don't have 'alt' attributes

### Usage

```
images_noalt_scrap(link, askRobot = FALSE)
```

### Arguments

| | |
|---|---|
| link | the URL of the web page |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

### Value

a character vector of images' URL without "alt" attribute

### Examples

```
images_noalt_scrap(link = "https://www.r-consortium.org/")
```

## images_preview *Scrape Images URLs*

### Description

Scrape Images URLs

### Usage

```
images_preview(link, askRobot = FALSE)
```

### Arguments

| | |
|---|---|
| link | the link of the web page |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

## Value

Images URLs

## Examples

```
images_preview(link = "https://posit.co/")
```

---

`images_scrap` *Scrape Images from a Web Page*

---

## Description

Scrape Images from a Web Page

## Usage

```
images_scrap(link, imgpath = getwd(), extn, askRobot = FALSE)
```

## Arguments

| | |
|---|---|
| `link` | the link of the web page |
| `imgpath` | the path of the images. Defaults to the current directory |
| `extn` | the extension of the image: png, jpeg ... |
| `askRobot` | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

## Value

called for the side effect of downloading images

## Examples

```
## Not run:

images_scrap(link = "https://posit.co/", extn = "jpg")


## End(Not run)
```

paragraphs_scrap *Website text paragraph scraping*

## Description

This function is used to scrape text paragraphs from a website.

## Usage

```
paragraphs_scrap(
  link,
  contain = NULL,
  case_sensitive = FALSE,
  collapse = FALSE,
  askRobot = FALSE
)
```

## Arguments

| | |
|---|---|
| link | the link of the web page to scrape |
| contain | filter the paragraphs according to the character string provided. |
| case_sensitive | logical. Should the contain argument be case sensitive ? defaults to FALSE |
| collapse | if TRUE the paragraphs will be collapsed into one element and the contain argument ignored. |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrap the web page ? Default is FALSE. |

## Value

a character vector.

## Examples

```
# Extracting the paragraphs displayed on the health page of the New York Times

link     <- "https://www.nytimes.com/section/health"

paragraphs_scrap(link)
```

## pdf_scrap *Scrape and download pdf files from a Web Page*

### Description

Scrape and download pdf files from a Web Page

### Usage

```
pdf_scrap(link, path = getwd(), askRobot = FALSE)
```

### Arguments

| | |
|---|---|
| link | the link of the web page |
| path | the path where to save the PDF files. Defaults to the current directory |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

### Value

called for the side effect of downloading PDF files from a website

## scrap *Simple website scraping*

### Description

This function is used to scrape one element from a website.

### Usage

```
scrap(link, node, clean = FALSE, askRobot = FALSE)
```

### Arguments

| | |
|---|---|
| link | the link of the web page to scrape |
| node | the HTML or CSS element to consider, the SelectorGadget tool is highly recommended |
| clean | logical. Should the function clean the extracted vector or not ? Default is FALSE. |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

## Value

a character vector

## Examples

```
# Extracting imdb top 250 movie titles
  link <- "https://www.imdb.com/chart/top/"
  node <- "h3.ipc-title__text"
  scrap(link, node)
```

---

`table_scrap` *HTML table scraping*

---

## Description

This function is used to scrape an html table from a website.

## Usage

```
table_scrap(link, choose = 1, header = TRUE, askRobot = FALSE)
```

## Arguments

| | |
|---|---|
| `link` | the link of the web page containing the table to scrape |
| `choose` | an integer indicating which table to scrape |
| `header` | do you want the first line to be the leader (default to TRUE) |
| `askRobot` | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

## Value

a data frame object.

## Examples

```
# Extracting premier ligue 2019/2020 top scorers

link <- "https://www.topscorersfootball.com/premier-league"
table_scrap(link)
```

## tidy_scrap *Website Tidy scraping*

### Description

This function is used to scrape a tibble from a website.

### Usage

```
tidy_scrap(link, nodes, colnames, clean = FALSE, askRobot = FALSE)
```

### Arguments

| | |
|---|---|
| link | the link of the web page to scrape |
| nodes | the vector of HTML or CSS elements to consider, the SelectorGadget tool is highly recommended. |
| colnames | the names of the expected columns. |
| clean | logical. Should the function clean the extracted tibble or not ? Default is FALSE. |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

### Value

a tidy data frame.

### Examples

```
# Extracting imdb movie titles and rating
link     <- "https://www.imdb.com/chart/top/"
my_nodes <- c("a > h3.ipc-title__text", "span.ratingGroup--imdb-rating")
names    <- c("title", "rating")
tidy_scrap(link, my_nodes, names)
```

## titles_scrap *Website title scraping*

### Description

This function is used to scrape titles (h1, h2 & h3 html tags) from a website. Useful for scraping daily electronic newspapers' titles.

### Usage

```
titles_scrap(link, contain = NULL, case_sensitive = FALSE, askRobot = FALSE)
```

### Arguments

| | |
|---|---|
| link | the link of the web page to scrape |
| contain | filter the titles according to a character string provided. |
| case_sensitive | logical. Should the contain argument be case sensitive ? defaults to FALSE |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE |

### Value

a character vector

### Examples

```
# Extracting the current titles of the New York Times

link     <- "https://www.nytimes.com/"

titles_scrap(link)
```

---

## weblink_scrap *Website web links scraping*

---

### Description

This function is used to scrape web links from a website.

### Usage

```
weblink_scrap(link, contain = NULL, case_sensitive = FALSE, askRobot = FALSE)
```

### Arguments

| | |
|---|---|
| link | the link of the web page to scrape |
| contain | filter the web links according to the character string provided. |
| case_sensitive | logical. Should the contain argument be case sensitive ? defaults to FALSE |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

### Value

a character vector.

## Examples

```
# Extracting the web links within the World Bank research and publications page

link <- "https://www.worldbank.org/en/research"

weblink_scrap(link)
```

---

xls_scrap *Scrape and download Excel xls files from a Web Page*

---

## Description

Scrape and download Excel xls files from a Web Page

## Usage

```
xls_scrap(link, path = getwd(), askRobot = FALSE)
```

## Arguments

| | |
|---|---|
| link | the link of the web page |
| path | the path where to save the Excel xls files. Defaults to the current directory |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

## Value

called for the side effect of downloading Excel xls files from a website

---

xlsx_scrap *Scrape and download Excel xlsx files from a Web Page*

---

## Description

Scrape and download Excel xlsx files from a Web Page

## Usage

```
xlsx_scrap(link, path = getwd(), askRobot = FALSE)
```

## Arguments

| | |
|---|---|
| link | the link of the web page |
| path | the path where to save the Excel xlsx files. Defaults to the current directory |
| askRobot | logical. Should the function ask the robots.txt if we're allowed or not to scrape the web page ? Default is FALSE. |

## Value

called for the side effect of downloading Excel xlsx files from a website

## Examples

```
## Not run:

xlsx_scrap(
link = "https://www.rieter.com/investor-relations/results-and-presentations/financial-statements"
)


## End(Not run)
```

# Index

attribute_scrap, 2
audio_scrap, 3

colors_scrap, 4
comments_scrap, 4
csv_scrap, 5

images_noalt_scrap, 6
images_preview, 6
images_scrap, 7

paragraphs_scrap, 8
pdf_scrap, 9

scrap, 9

table_scrap, 10
tidy_scrap, 11
titles_scrap, 11

weblink_scrap, 12

xls_scrap, 13
xlsx_scrap, 13