Package 'urlexplorer'

May 8, 2026

Title Structural Analysis and Pattern Discovery in URL Datasets

Version 0.1.0

Description Offers tools for parsing and analyzing URL datasets, extracting key components and identifying common patterns. It aids in examining website architecture and identifying SEO issues, helping users optimize web presence and content strategy.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports dplyr, rlang, stringr, tibble, tidyr

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://github.com/MarekProkop/urlexplorer

BugReports https://github.com/MarekProkop/urlexplorer/issues

Depends R (>= 4.1.0)

LazyData true

NeedsCompilation no

Author Marek Prokop [aut, cre]

Maintainer Marek Prokop <mprokop@prokopsw.cz>

Repository CRAN

Date/Publication 2025-07-14 16:40:02 UTC

Contents

`count_fragments` *Count fragments in URLs*

Description

Count fragments in URLs

Usage

```
count_fragments(url, sort = FALSE, name = "n")
```

Arguments

`url`	A character vector of URLs.
`sort`	Logical indicating whether to sort the output by count. Defaults to FALSE.
`name`	The name of the column containing the counts. Defaults to 'n'.

Value

A tibble with each fragment and its count.

Examples

```
count_fragments(c("http://example.com#top", "http://example.com#bottom"))
```

count_hosts *Count different hosts found in URLs*

Description

Count different hosts found in URLs

Usage

```
count_hosts(url, sort = FALSE, name = "n")
```

Arguments

url	A character vector of URLs.
sort	Logical indicating whether to sort the output by count. Defaults to FALSE.
name	The name of the column containing the counts. Defaults to 'n'.

Value

A tibble with each host and its count.

Examples

```
count_hosts(c("http://example.com", "http://www.example.com"))
```

count_param_names *Count different parameter names in query strings*

Description

Count different parameter names in query strings

Usage

```
count_param_names(query, sort = FALSE, name = "n")
```

Arguments

query	A character vector of query strings.
sort	Logical indicating whether to sort the output by count. Defaults to FALSE.
name	The name of the column containing the counts. Defaults to 'n'.

Value

A tibble with each parameter name and how often it occurs.

Examples

```
count_param_names(c("param1=value1&param2=value2", "param3=value3"))
```

`count_param_values` *Count different values for a specified parameter across query strings*

Description

Count different values for a specified parameter across query strings

Usage

```
count_param_values(query, param_name, sort = FALSE, name = "n")
```

Arguments

`query`	A character vector of query strings.
`param_name`	The name of the parameter whose values to count.
`sort`	Logical indicating whether to sort the output by count. Defaults to FALSE.
`name`	The name of the column containing the counts. Defaults to 'n'.

Value

A tibble with each value of the specified parameter and how often it occurs.

Examples

```
count_param_values(c("param1=value1&param2=value2", "param1=value3"), "param1")
```

`count_paths` *Count different paths found in URLs*

Description

Count different paths found in URLs

Usage

```
count_paths(url, sort = FALSE, name = "n")
```

Arguments

`url`	A character vector of URLs.
`sort`	Logical indicating whether to sort the output by count. Defaults to FALSE.
`name`	The name of the column containing the counts. Defaults to 'n'.

Value

A tibble with each path and its count.

Examples

```
count_paths(c("http://example.com/index", "http://example.com/home"))
```

`count_path_segments` *Count occurrences of specific path segments at a given index*

Description

Count occurrences of specific path segments at a given index

Usage

```
count_path_segments(path, segment_index, sort = FALSE, name = "n")
```

Arguments

`path`	A character vector of paths.
`segment_index`	Index of the segment to count.
`sort`	Logical indicating whether to sort the output by count. Defaults to FALSE.
`name`	The name of the column containing the counts. Defaults to 'n'.

Value

A tibble with each segment at the specified index and how often it occurs.

Examples

```
count_path_segments(c("/path/to/resource", "/path/to/shop"), 2)
```

count_ports *Count different port numbers used in URLs*

Description

Count different port numbers used in URLs

Usage

```
count_ports(url, sort = FALSE, name = "n")
```

Arguments

url	A character vector of URLs.
sort	Logical indicating whether to sort the output by count. Defaults to FALSE.
name	The name of the column containing the counts. Defaults to 'n'.

Value

A tibble with each port and how many times it occurs.

Examples

```
count_ports(c("http://example.com:8080", "http://example.com:80"))
```

count_queries *Count the occurrence of query strings in URLs*

Description

Count the occurrence of query strings in URLs

Usage

```
count_queries(url, sort = FALSE, name = "n")
```

Arguments

url	A character vector of URLs.
sort	Logical indicating whether to sort the output by count. Defaults to FALSE.
name	The name of the column containing the counts. Defaults to 'n'.

Value

A tibble with each query string and how often it occurs.

Examples

```
count_queries(c("http://example.com?query1=value1", "http://example.com?query2=value2"))
```

---

count_schemes *Count different schemes used in URLs*

Description

Count different schemes used in URLs

Usage

```
count_schemes(url, sort = FALSE, name = "n")
```

Arguments

url	A character vector of URLs.
sort	Logical indicating whether to sort the output by count. Defaults to FALSE.
name	The name of the column containing the counts. Defaults to 'n'.

Value

A tibble with each scheme and its count.

Examples

```
count_schemes(c("http://example.com", "https://example.com"))
```

---

count_userinfos *Count occurrences of userinfo in URLs*

Description

Count occurrences of userinfo in URLs

Usage

```
count_userinfos(url, sort = FALSE, name = "n")
```

Arguments

url	A character vector of URLs.
sort	Logical indicating whether to sort the output by count. Defaults to FALSE.
name	The name of the column containing the counts. Defaults to 'n'.

Value

A tibble listing userinfos and how often each occurs.

Examples

```
count_userinfos(c("http://user:pass@example.com", "http://example.com"))
```

`extract_file_extension` *Extract file extension from URLs or paths*

Description

This function parses each input URL or path and extracts the file extension, if present. It is particularly useful for identifying the type of files referenced in URLs.

Usage

```
extract_file_extension(url)
```

Arguments

`url`	A character vector of URLs or paths from which to extract file extensions.

Value

A character vector with the file extension for each URL or path. Extensions are returned without the dot (e.g., "jpg" instead of ".jpg"), and URLs or paths without extensions will return `NA`.

Examples

```
extract_file_extension(
  c(
    "http://example.com/image.jpg",
    "https://example.com/archive.zip",
    "http://example.com/"
  )
)
```

extract_fragment *Extract the fragment from URL*

Description

Extract the fragment from URL

Usage

```
extract_fragment(url)
```

Arguments

url	A character vector of URLs.

Value

A character vector containing the fragment from each URL, if present.

Examples

```
extract_fragment(c("http://example.com/#sec1", "http://example.com/#sec2"))
```

extract_host *Extract the host from URL*

Description

Extract the host from URL

Usage

```
extract_host(url)
```

Arguments

url	A character vector of URLs.

Value

A character vector containing the host from each URL.

Examples

```
extract_host(c("https://example.com", "http://www.example.com"))
```

extract_param_value *Extract the value of a specified parameter from the query string*

Description

Extract the value of a specified parameter from the query string

Usage

```
extract_param_value(query, param_name)
```

Arguments

query	A character vector of query strings.
param_name	The name of the parameter to extract values for.

Value

A character vector containing the value of the specified parameter from each query string.

Examples

```
extract_param_value(c("param1=val1&param2=val2", "param1=val3"), "param1")
```

extract_path *Extract the path from URL*

Description

Extract the path from URL

Usage

```
extract_path(url)
```

Arguments

url	A character vector of URLs.

Value

A character vector containing the path from each URL.

Examples

```
extract_path(c("http://example.com/", "http://example.com/path/to/resource"))
```

extract_path_segment *Extract a specific segment from a path*

Description

Extract a specific segment from a path

Usage

```
extract_path_segment(path, segment_index)
```

Arguments

path	A character vector of paths.
segment_index	The index of the segment to extract.

Value

A character vector containing the specified segment from each path.

Examples

```
extract_path_segment(c("/path/to/resource", "/another/path/"), 2)
```

extract_port *Extract the port number from URL*

Description

Extract the port number from URL

Usage

```
extract_port(url)
```

Arguments

url	A character vector of URLs.

Value

A character vector containing the port number from each URL, if specified.

Examples

```
extract_port(c("http://example.com:8080"))
```

extract_query *Extract the query from URL*

Description

Extract the query from URL

Usage

```
extract_query(url)
```

Arguments

url	A character vector of URLs.

Value

A character vector containing the query string from each URL.

Examples

```
extract_query(c(
  "http://example.com?query1=value1&query2=value2",
  "http://example.com?query1=value3"
))
```

extract_scheme *Extract the scheme from URL*

Description

Extract the scheme from URL

Usage

```
extract_scheme(url)
```

Arguments

url	A character vector of URLs.

Value

A character vector containing the scheme from each URL.

Examples

```
extract_scheme(c("http://example.com", "https://example.com"))
```

extract_userinfo *Extract userinfo from URL*

Description

Extract userinfo from URL

Usage

```
extract_userinfo(url)
```

Arguments

url	A character vector of URLs.

Value

A character vector containing the userinfo from each URL, if present.

Examples

```
extract_userinfo(c("http://user:pass@example.com"))
```

split_host *Split host into subdomains and domain*

Description

Split host into subdomains and domain

Usage

```
split_host(host)
```

Arguments

host	A character vector of hostnames to be split.

Value

A tibble with one row per hostname and columns for top-level domain, domain and subdomains. Columns are created as many as the number of hosts' components and are named as tld, domain, subdomain_1, subdomain_2, etc.

Examples

```
split_host(c("subdomain.example.com"))
split_host(c("subdomain2.subdomain1.example.com", "example.com"))
```

split_path *Split path into segments*

Description

Split path into segments

Usage

```
split_path(path)
```

Arguments

path	A character vector of paths to be split.

Value

A tibble with one row per path and columns for each segment separated by '/'.

Examples

```
split_path(c("/path/to/resource"))
```

split_query *Split query into parameters*

Description

Split query into parameters

Usage

```
split_query(query)
```

Arguments

query	A character vector of query strings to be split.

Value

A tibble with one row per query string and columns for each parameter, column names as parameter names.

Examples

```
split_query(c("param1=value1&param2=value2"))
```

split_url *Split URL into its constituent parts*

Description

Split URL into its constituent parts

Usage

```
split_url(url)
```

Arguments

url	A character vector of URLs to be split.

Value

A tibble with one row per URL and columns for each component: scheme, host, port, userinfo, path, query, and fragment.

Examples

```
split_url(c("https://example.com/path?query=arg#frag"))
```

websitepages *Sample web site URLs*

Description

Sample web site URLs

Usage

```
websitepages
```

Format

`websitepages`**:**
A data frame with 1,000 rows and 1 column:

page Page URL ...

Source

Syntetic data

Index