Package: waetr (via r-universe)

October 10, 2026

Title 'WebAIM' 'WAVE' Accessibility Evaluation Tool

Version 0.1.0

Description An R interface to the 'WebAIM' 'WAVE' accessibility evaluation API <https://wave.webaim.org/api/>. This package provides tools for analyzing web pages for accessibility issues, generating reports, and comparing accessibility across multiple websites.

License MIT + file LICENSE

URL https://github.com/benjaminlistyg/waetr

BugReports https://github.com/benjaminlistyg/waetr/issues

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Imports dplyr, ggplot2, httr, jsonlite, purrr (>= 0.3.0), tibble, tools

Suggests mockery, testthat (>= 3.0.0)

LazyData true

Config/testthat/edition 3

Config/pak/sysreqs libssl-dev

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

Date/Publication 2025-09-03 16:24:45 UTC

RemoteUrl https://github.com/benjaminlistyg/waetr

RemoteRef HEAD

RemoteSha fecc097ecdabe81e5d73795279b55fc8cf3bd0ae

Contents

check_wave_credits 2
compare_accessibility 2

create_accessibility_report 3
fetch_wave_data 4
wave 5

Index **7**

`check_wave_credits` *Check WAVE API Credits*

Description

Checks remaining WAVE API credits for the provided key

Usage

```
check_wave_credits(api_key)
```

Arguments

`api_key`	Character string. WAVE API key

Value

Numeric value of remaining credits

Examples

```
## Not run:
credits <- check_wave_credits("your_api_key")
print(sprintf("Remaining credits: %d", credits))

## End(Not run)
```

`compare_accessibility` *Compare Website Accessibility*

Description

Main function to generate accessibility comparisons across multiple websites

Usage

```
compare_accessibility(
  input,
  api_key = NULL,
  site_names = NULL,
  plot_type = c("category_counts", "issues", "structure"),
  report_type = 1,
  theme = "light"
)
```

Arguments

input	Character vector. URLs to analyze or paths to JSON files
api_key	Character string. WAVE API key (required for URL analysis)
site_names	Character vector. Optional custom names for sites
plot_type	Character string. Type of visualization: • "category_counts": Compare main accessibility categories • "issues": Detailed breakdown of errors and alerts • "structure": Compare structural elements
report_type	Integer. WAVE report type (1-4)
theme	Character string. Visual theme for plot (default: "light")

Value

ggplot object containing the requested visualization

Examples

```
## Not run:
# Compare multiple websites
p <- compare_accessibility(
  input = c("https://example.com", "https://example.org"),
  api_key = "your_api_key",
  plot_type = "category_counts"
)

# Save the plot to a temporary directory
ggsave(file.path(tempdir(), "accessibility_comparison.png"), p)

## End(Not run)
```

---

create_accessibility_report *Create Comprehensive Accessibility Report*

Description

Generates a complete accessibility report with visualizations and summary data

Usage

```
create_accessibility_report(
  input,
  api_key = NULL,
  output_dir = NULL,
  report_type = 1,
  include_plots = TRUE,
  custom_theme = "light"
)
```

Arguments

input	Character vector. URLs to analyze or paths to JSON files
api_key	Character string. WAVE API key (required for URL analysis)
output_dir	Character string. Directory to save report files
report_type	Integer. WAVE report type (1-4)
include_plots	Logical. Whether to include plots in the report (default: TRUE)
custom_theme	Character string. Visual theme for plots (default: "light")

Value

List containing plots and summary data frame

Examples

```
## Not run:
report <- create_accessibility_report(
  input = c("https://example.com", "https://example.org"),
  api_key = "your_api_key",
  output_dir = tempdir()
)

## End(Not run)
```

---

fetch_wave_data *Fetch WAVE Analysis Data*

Description

Retrieves accessibility analysis data either from WAVE API or local JSON files

Usage

```
fetch_wave_data(
  input,
  api_key = NULL,
  report_type = 1,
  is_json = FALSE,
  delay = 1
)
```

Arguments

input	Character vector. Either URLs to analyze or paths to JSON files
api_key	Character string. WAVE API key (required for URL analysis)
report_type	Integer. WAVE report type (1-4): • 1: Basic statistics only (1 credit) • 2: Includes WAVE items (2 credits) • 3: Includes XPath data (3 credits) • 4: Includes CSS selector data (3 credits)
is_json	Logical. Whether input contains JSON file paths (default: FALSE)
delay	Numeric. Delay between API calls in seconds (default: 1)

Value

List of WAVE analysis results

Examples

```
## Not run:
# Fetch from URLs
results <- fetch_wave_data(
  input = c("https://example.com", "https://example.org"),
  api_key = "your_api_key"
)

# Load from JSON files
json_results <- fetch_wave_data(
  input = c("site1.json", "site2.json"),
  is_json = TRUE
)

## End(Not run)
```

---

wave *Access WebAIM WAVE Accessibility API*

Description

This function provides an interface to the WebAIM WAVE accessibility evaluation API. It allows you to analyze web pages for accessibility issues and retrieve detailed reports.

Usage

```
wave(
  key,
  url,
  format = NULL,
  viewportwidth = NULL,
  reporttype = NULL,
  username = NULL,
  password = NULL,
  useragent = NULL,
  toDataframe = FALSE,
  file = NULL
)
```

Arguments

key	Character string. Your WAVE API key
url	Character string. URL of the webpage to analyze
format	Character string. Response format (optional)
viewportwidth	Integer. Viewport width for analysis (optional)
reporttype	Integer. Type of report to generate (1-4) (optional)
username	Character string. Username for protected pages (optional)
password	Character string. Password for protected pages (optional)
useragent	Character string. Custom user agent (optional)
toDataframe	Logical. Whether to convert results to a data frame (default: FALSE)
file	Character string. Optional file path to save JSON results

Value

List or tibble containing WAVE analysis results

Examples

```
## Not run:
# Basic usage
results <- wave(key = "your_api_key", url = "https://example.com")

# Get results as a data frame
df_results <- wave(key = "your_api_key", url = "https://example.com", toDataframe = TRUE)

# Save results to a temporary file
tmp_file <- file.path(tempdir(), "wave_results.json")
wave(key = "your_api_key", url = "https://example.com", file = tmp_file)

## End(Not run)
```

Index

check_wave_credits, 2
compare_accessibility, 2
create_accessibility_report, 3

fetch_wave_data, 4

wave, 5