Package: wasserportal (via r-universe)

June 6, 2026

Title R Package with Functions for Scraping Data of Wasserportal Berlin

Version 0.5.0

Description R Package with Functions for Scraping Data of Wasserportal Berlin (https://wasserportal.berlin.de), which contains real-time data of surface water and groundwater monitoring stations.

License MIT + file LICENSE

URL https://github.com/KWB-R/wasserportal

BugReports https://github.com/KWB-R/wasserportal/issues

Imports archive, data.table, dplyr, fs, httr, kwb.datetime, kwb.utils, magrittr, readr, rlang, rvest, stringr, tibble, tidyr, withr, xml2

Suggests covr, DT, forcats, ggplot2, gridExtra, htmltools, htmlwidgets, janitor, jsonlite, knitr, kwb.pkgbuild, leaflet, openxlsx, plotly, rmarkdown, sf, testthat (>= 3.0.0), tidyselect

VignetteBuilder knitr

Remotes github::kwb-r/kwb.datetime, github::kwb-r/kwb.pkgbuild, github::kwb-r/kwb.utils

Config/testthat/edition 3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Config/pak/sysreqs cmake make libarchive-dev libicu-dev libuv1-dev libxml2-dev libssl-dev libx11-dev

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

Date/Publication 2026-05-07 18:02:56 UTC

RemoteUrl https://github.com/KWB-R/wasserportal

RemoteRef HEAD

RemoteSha a5ecad7ed209690907d4c37adfe1e40f115b0982

Contents

`base_url_download` *Helper function: base url for download*

Description

Helper function: base url for download

Usage

```
base_url_download()
```

Value

base url for download of csv/zip files prepared by R package

---

columns_to_labels *Create Text Labels from Data Frame Columns*

Description

Create Text Labels from Data Frame Columns

Usage

```
columns_to_labels(data, columns, fmt = "%s: %s", sep = ", ")
```

Arguments

data	data frame
columns	names of columns from which to create labels
fmt	format string passed to sprintf
sep	separator (default: ", ")

Value

vector of character with as many elements as there are rows in data

Examples

```
data <- data.frame(number = 1:2, name = c("adam", "eva"), value = 3:4)
columns <- c("name", "value")
columns_to_labels(data, columns)
columns_to_labels(data, columns, fmt = "<p>%s: %s</p>", sep = "")
```

---

get_api_tables *Provide Tables of Wasserportal API Documentation*

Description

The tables that appear in the API documentation of the wasserportal (https://wasserportal.berlin.de/download/wasserportal_be
have been added to the wasserportal package. This function returns a list of data frames with each
element representing one of these tables.

Usage

```
get_api_tables(name = NULL)
```

Arguments

name	of element from the list of data frames to be selected. If this argument is left blank (name = NULL), the default, the list of data frames is returned.

Value

list of data frames or data frame specified by the name argument

Examples

```
get_api_tables()
```

---

get_daily_surfacewater_data

Get Daily Surfacewater Data: wrapper to scrape daily surface water data

---

Description

Get Daily Surfacewater Data: wrapper to scrape daily surface water data

Usage

```
get_daily_surfacewater_data(
  stations,
  variables = get_surfacewater_variables(),
  list2df = FALSE
)
```

Arguments

stations	stations as retrieved by by get_stations
variables	variables as retrieved by by get_surfacewater_variables
list2df	convert result list to data frame (default: FALSE)

Value

list or data frame with all available data from Wasserportal

Examples

```
## Not run:
stations <- wasserportal::get_stations()
variables <- wasserportal::get_surfacewater_variables()
variables
sw_data_daily <- wasserportal::get_daily_surfacewater_data(stations, variables)

## End(Not run)
```

`get_groundwater_data` *Get Groundwater Data*

Description

wrapper function to scrape all available raw data, i.e. groundwater level and quality data and save in list

Usage

```
get_groundwater_data(
  stations,
  groundwater_options = get_groundwater_options(),
  debug = TRUE,
  stations_list = NULL
)
```

Arguments

`stations`	list as retrieved by `get_stations`. Deprecated. Please use `stations_list` instead
`groundwater_options`	as retrieved by `get_groundwater_options`
`debug`	print debug messages (default: TRUE)
`stations_list`	list of station metadata as returned by `get_stations(type = "list")`

Value

list with elements "groundwater.level" and "groundwater.quality" data frames

Examples

```
## Not run:
stations <- wasserportal::get_stations()
gw_data_list <- get_groundwater_data(stations)
str(gw_data_list)

## End(Not run)
```

`get_groundwater_options` *Helper function: get groundwater options*

Description

Helper function: get groundwater options

Usage

```
get_groundwater_options()
```

Value

return available groundwater data options and prepare for being used as input for `get_groundwater_data`

Examples

```
get_groundwater_options()
```

`get_overview_options` *Wasserportal Berlin: get overview options for stations*

Description

Wasserportal Berlin: get overview options for stations

Usage

```
get_overview_options()
```

Value

list with shortcuts to station overview tables (`wasserportal.berlin.de/messwerte.php?anzeige=tabelle&thema=<shor`

Examples

```
get_overview_options()
```

get_station_variables *Helper function: get available station variables*

Description

Helper function: get available station variables

Usage

```
get_station_variables(station_df)
```

Arguments

station_df	data frame with one row per station and columns "Messstellennummer", "Messstellenname" and additional columns each of which represents a variable that is measured at that station. If the variable columns contain the value "x" it means that the corresponding variable is measured and the name of the column is contained in the returned vector of variable names.

Value

returns names of available variables for station

get_stations *Get Stations*

Description

Get Stations

Usage

```
get_stations(
  type = c("list", "data.frame", "crosstable"),
  run_parallel = TRUE,
  n_cores = parallel::detectCores() - 1L,
  debug = TRUE
)
```

Arguments

type	vector of character describing the type(s) of output(s) to be returned. Expected values (and default): `c("list", "data.frame", "crosstable")`. If only one value is given the data is returned in the expected type. If more than one values are given, a list is returned with one list element per type.
run_parallel	default: TRUE
n_cores	number of cores to use if `run_parallel = TRUE`. Default: one less than the detected number of cores.
debug	logical indicating whether or not to show debug messages

Value

list with general station "overview" (either as list "overview_list" or as data.frame "overview_df") and a crosstable with information which parameters is available per station ("x" if available, NA if not)

Examples

```
stations <- wasserportal::get_stations(n_cores = 2L)
str(stations)
```

---

get_surfacewater_qualities

Get Surface Water Quality for Multiple Monitoring Stations

Description

Get Surface Water Quality for Multiple Monitoring Stations

Usage

```
get_surfacewater_qualities(station_ids, dbg = TRUE)
```

Arguments

station_ids	vector with ids of multiple (or one) monitoring stations
dbg	print debug messages (default: TRUE)

Value

data frame with water quality data for multiple monitoring stations

Examples

```
## Not run:
stations <- wasserportal::get_stations()
station_ids <- stations$overview_list$surface_water.quality$Messstellennummer
swq <- wasserportal::get_surfacewater_qualities(station_ids)
str(swq)

## End(Not run)
```

`get_surfacewater_quality` *Get Surface Water Quality for One Monitoring Station*

Description

Get Surface Water Quality for One Monitoring Station

Usage

```
get_surfacewater_quality(station_id)
```

Arguments

`station_id`	id of surface water measurement station

Value

data frame with water quality data for one monitoring station

Examples

```
## Not run:
stations <- wasserportal::get_stations()
station_id <- stations$overview_list$surface_water.quality$Messstellennummer[1]
swq <- wasserportal::get_surfacewater_quality(station_id)
str(swq)

## End(Not run)
```

`get_surfacewater_variables` *Helper function: get surface water variables*

Description

Helper function: get surface water variables

Usage

```
get_surfacewater_variables()
```

Value

vector with surface water variables

`get_wasserportal_master_data` *Wasserportal Berlin: get master data for a single station*

Description

Wasserportal Berlin: get master data for a single station

Usage

```
get_wasserportal_master_data(master_url)
```

Arguments

`master_url`	url with master data for single station as retrieved by `get_wasserportal_stations_table`

Value

data frame with metadata for selected station

Examples

```
## Not run:
stations_list <- wasserportal::get_stations(type = "list")

# GW Station
master_url <- stations_list %>%
  kwb.utils::selectElements("groundwater.level") %>%
  kwb.utils::selectColumns("stammdaten_link")[1L]

get_wasserportal_master_data(master_url)
```

```
# SW Station

# Reduce  to monitoring stations maintained by Berlin
master_urls <- stations_list %>%
  kwb.utils::selectElements("surface_water.water_level") %>%
  dplyr::filter(.data$Betreiber == "Land Berlin") %>%
  dplyr::pull(.data$stammdaten_link)

get_wasserportal_master_data(master_urls[1L])

## End(Not run)
```

`get_wasserportal_masters_data` *Wasserportal Berlin: get master data for a multiple stations*

Description

Wasserportal Berlin: get master data for a multiple stations

Usage

```
get_wasserportal_masters_data(master_urls, run_parallel = TRUE)
```

Arguments

`master_urls`	URLs to master data as found in column "stammdaten_link" of the data frame returned by `get_stations(type = "list")`
`run_parallel`	default: TRUE

Value

data frame with metadata for selected master urls

Examples

```
## Not run:
stations_list <- wasserportal::get_stations(type = "list")

# Reduce  to monitoring stations maintained by Berlin
master_urls <- stations_list$surface_water.water_level %>%
  dplyr::filter(.data$Betreiber == "Land Berlin") %>%
  dplyr::pull(.data$stammdaten_link)

system.time(master_parallel <- get_wasserportal_masters_data(
  master_urls
))
```

```
system.time(master_sequential <- get_wasserportal_masters_data(
  master_urls,
  run_parallel = FALSE
))

## End(Not run)
```

`get_wasserportal_stations` *Get Names and IDs of the Stations of wasserportal.berlin.de*

Description

Get Names and IDs of the Stations of wasserportal.berlin.de

Usage

```
get_wasserportal_stations(type = "quality")
```

Arguments

`type`	one of "quality", "level", "flow"

`get_wasserportal_stations_table` *Wasserportal Berlin: get stations overview table*

Description

Wasserportal Berlin: get stations overview table

Usage

```
get_wasserportal_stations_table(
  type = get_overview_options()$groundwater$level,
  url_wasserportal = wasserportal_base_url()
)
```

Arguments

`type`	type of stations table to retrieve. Valid options defined in `get_overview_options`, default: get_overview_options()$groundwater$level
`url_wasserportal`	base url to Wasserportal berlin (default: `wasserportal_base_url`

Value

data frame with master data of selected monitoring stations

Examples

```
types <- wasserportal::get_overview_options()
str(types)
sw_l <- wasserportal::get_wasserportal_stations_table(type = types$surface_water$water_level)
str(sw_l)
```

---

get_wasserportal_variables

Get Names and IDs of the Variables of wasserportal.berlin.de

Description

Get Names and IDs of the Variables of wasserportal.berlin.de

Usage

```
get_wasserportal_variables(station = NULL)
```

Arguments

station	station id. If given, only variables that are available for the given station are returned.

---

list_data_to_csv_or_zip

Helper function: list data to csv or zip

Description

Helper function: list data to csv or zip

Usage

```
list_data_to_csv_or_zip(data_list, file_prefix, to_zip)
```

Arguments

data_list	data in list form
file_prefix	file prefix
to_zip	whether or not to convert to zip file

Value

loops through list of data frames and uses list names as filenames

list_masters_data_to_csv

Helper function: list masters data to csv

Description

Helper function: list masters data to csv

Usage

```
list_masters_data_to_csv(masters_data_list)
```

Arguments

`masters_data_list`
: masters data in list form as retrieved by `get_stations(type = "list")`

Value

loops through list of data frames and uses list names as filenames

Examples

```
## Not run:
stations_list <- get_stations(type = "list")
masters_data_csv_files <- list_masters_data_to_csv(stations_list)
masters_data_csv_files

## End(Not run)
```

list_timeseries_data_to_zip

Helper function: list timeseries data to zip

Description

Helper function: list timeseries data to zip

Usage

```
list_timeseries_data_to_zip(timeseries_data_list)
```

Arguments

`timeseries_data_list`
: time series data in list form as retrieved by `get_groundwater_data`

Value

loops through list of data frames and uses list names as filenames

Examples

```
## Not run:
stations <- wasserportal::get_stations()

# Groundwater Time Series
gw_tsdata_list <- wasserportal::get_groundwater_data(stations)
gw_tsdata_files <- wasserportal::list_timeseries_data_to_zip(gw_tsdata_list)

# Surface Water Time Series
sw_tsdata_list <- wasserportal::get_daily_surfacewater_data(stations)
sw_tsdata_files <- wasserportal::list_timeseries_data_to_zip(sw_tsdata_list)

## End(Not run)
```

---

read *Helper function to read CSV*

Description

Helper function to read CSV

Usage

```
read(text, ...)
```

Arguments

text	text
...	... additional arguments passed to read.table

Value

data frame with values

read_wasserportal *Download and Read Data from wasserportal.berlin.de*

Description

This function downloads and reads CSV files from wasserportal.berlin.de.

Usage

```
read_wasserportal(
  station,
  variables = NULL,
  from_date = as.character(Sys.Date() - 90L),
  type = "single",
  include_raw_time = FALSE,
  stations_crosstable
)
```

Arguments

station	station number, as found in column "Messstellennummer" of the data frame returned by `get_stations(type = "crosstable")`
variables	vector of variable identifiers, as returned by `get_station_variables`
from_date	`Date` object (or string in format "yyyy-mm-dd" that can be converted to a `Date` object representing the first day for which to request data. Default: `as.character(Sys.Date() - 90L)`
type	one of "single" (the default), "daily", "monthly"
include_raw_time	if `TRUE` the original time column and the column with the corrected winter time are included in the output. The default is `FALSE`.
stations_crosstable	data frame as returned by `get_stations(type = "crosstable")`

Details

The original timestamps (column `timestamps_raw` in the example below) are not all plausible, e.g. "31.03.2019 03:00" appears twice! They are corrected (column `timestamp_corr`) to represent a plausible sequence of timestamps in Berlin Normal Time (UTC+01) Finally, a valid POSIXct timestamp in timezone "Berlin/Europe" (UTC+01 in winter, UTC+02 in summer) is created, together with the additional information on the UTC offset (column `UTCOffset`, 1 in winter, 2 in summer).

Value

data frame read from the CSV file that the download provides. IMPORTANT: It is not yet clear how to interpret the timestamp, see example

Examples

```
## Not run:
# Get a list of available water quality stations and variables
stations_crosstable <- wasserportal::get_stations(type = "crosstable")

# Set the start date
from_date <- "2021-03-01"

# Read the timeseries (multiple variables for one station)
water_quality <- wasserportal::read_wasserportal(
  station = stations_crosstable$Messstellennummer[1L],
  from_date = from_date,
  include_raw_time = TRUE,
  stations_crosstable = stations_crosstable
)

# Look at the first few records
head(water_quality)

# Check the metadata
#kwb.utils::getAttribute(water_quality, "metadata")

# Set missing values to NA
water_quality[water_quality == -777] <- NA

# Look at the first few records again
head(water_quality)

### How was the original timestamp interpreted?

# Determine the days at which summer time starts and ends, respectively
from_year <- as.integer(substr(from_date, 1L, 4L))
switches <- kwb.datetime::date_range_CEST(from_year)

# Reformat to dd.mm.yyyy
switches <- kwb.datetime::reformatTimestamp(switches, "%Y-%m-%d", "%d.%m.%Y")

# Define a pattern to look for timestamps "around" the switches
pattern <- paste(switches, "0[1-4]", collapse = "|")

# Look at the data for these timestamps
water_quality[grepl(pattern, water_quality$timestamp_raw), ]

# The original timestamps (timestamps_raw) were not all plausible, e.g.
# for March 2019. This seems to have been fixed by the "wasserportal"!
sum(water_quality$timestamp_raw != water_quality$timestamp_corr)

## End(Not run)
```

`read_wasserportal_raw` *Read Wasserportal Raw*

Description

Read Wasserportal Raw

Usage

```
read_wasserportal_raw(
  variable,
  station,
  from_date,
  type = "single",
  include_raw_time = FALSE,
  handle = NULL,
  stations_crosstable,
  api_version = 2L
)
```

Arguments

variable	variable
station	station id
from_date	start date
type	one of "single", "daily", "monthly" (default: "single")
include_raw_time	TRUE or FALSE (default: FALSE)
handle	handle (default: NULL)
stations_crosstable	data frame as returned by `get_stations(type = "crosstable")`
api_version	1 integer number representing the version of wasserportal's API. 1L: before 2023, 2L: since 2023. Default: 2L

Value

????

`read_wasserportal_raw_gw` *read_wasserportal_raw_gw*

Description

read_wasserportal_raw_gw

Usage

```
read_wasserportal_raw_gw(
  station = 149,
  stype = "gws",
  type = "single_all",
  from_date = "",
  include_raw_time = FALSE,
  handle = NULL,
  as_text = FALSE,
  dbg = FALSE
)
```

Arguments

station	station id
stype	"gws" or "gwq"
type	"single" or "single_all" (if stype = "gwq")
from_date	(default: "")
include_raw_time	default: FALSE
handle	default: NULL
as_text	if TRUE, the raw text that is returned by the HTTP request to the Wasserportal is returned by this function. Otherwise (the default) the raw text is tried to be interpreted as comma separated values and a corresponding data frame is returned. Use as_text = TRUE to analyse the raw text in case that an error occurs when trying to convert the text to a data frame.
dbg	logical indicating whether or not to show debug messages. The default is FALSE

Value

data.frame with values

Examples

```
## Not run:
read_wasserportal_raw_gw(station = 149, stype = "gws")
read_wasserportal_raw_gw(station = 149, stype = "gwq")

## End(Not run)
```

readPackageFile *Read CSV File from Package's "extdata" Folder*

Description

Read CSV File from Package's "extdata" Folder

Usage

```
readPackageFile(file, ...)
```

Arguments

file	file name (without path)
...	additional arguments passed to read.csv

Value

data frame representing the content of file

wasserportal_base_url *Helper function: Base Url of Berlin Wassersportal*

Description

Helper function: Base Url of Berlin Wassersportal

Usage

```
wasserportal_base_url()
```

Value

string with base url of Berlin Wasserportal

wp_masters_data_to_list *Wasserportal Master Data: download and Import in R List*

Description

Wasserportal Master Data: download and Import in R List

Usage

```
wp_masters_data_to_list(
  overview_list_names,
  target_dir = tempdir(),
  file_prefix = "stations_",
  is_zipped = FALSE
)
```

Arguments

overview_list_names	names of elements in the list returned by `get_stations(type = "list")`
target_dir	target directory for downloading data (default: tempdir())
file_prefix	prefix given to file names
is_zipped	are the data to be downloaded zipped (default: FALSE)

Value

downloads csv master data from Wasserportal

Examples

```
## Not run:
overview_list_names <- names(wasserportal::get_stations(type = "list"))
wp_masters_data_list <- wp_masters_data_to_list(overview_list_names)

## End(Not run)
```

`wp_timeseries_data_to_list` *Wasserportal Time Series Data: download and Import in R List*

Description

Wasserportal Time Series Data: download and Import in R List

Usage

```
wp_timeseries_data_to_list(
  overview_list_names,
  target_dir = tempdir(),
  is_zipped = TRUE
)
```

Arguments

`overview_list_names`	names of elements in the list returned by `get_stations(type = "list")`
`target_dir`	target directory for downloading data (default: tempdir())
`is_zipped`	are the data to be downloaded zipped (default: TRUE)

Value

downloads (zipped) data from wasserportal

Examples

```
## Not run:
overview_list_names <- names(wasserportal::get_stations(type = "list"))
wp_timeseries_data_list <- wp_timeseries_data_to_list(overview_list_names)

## End(Not run)
```

Index