Package 'hyd1d'

August 19, 2026

Type Package

Version 0.5.5

Date 2026-08-11

Title 1d Water Level Interpolation along the Rivers Elbe and Rhine

Description An S4 class and several functions which utilize internally stored datasets and gauging data enable 1d water level interpolation. The S4 class (WaterLevelDataFrame) structures the computation and visualisation of 1d water level information along the German federal waterways Elbe and Rhine. 'hyd1d' delivers 1d water level data - extracted from the 'FLYS' database - and validated gauging data - extracted from the hydrological database 'WISKI7' - package-internally. For computations near real time gauging data are queried externally from the 'PEGELONLINE REST API' <https://pegelonline.wsv.de/webservice/dokuRestapi>.

Depends R (>= 4.1.0)

Imports methods, utils, Rdpack, httr2, curl

Suggests DBI (>= 0.4-9), RPostgreSQL (>= 0.6-1), testthat, knitr, rmarkdown, stringr, devtools, pak, pkgdown, roxygen2, revealjs, shiny, shiny.i18n, shinyTime, lubridate, usethis, yaml, desc, rsconnect

RdMacros Rdpack

License GPL (>= 2)

Encoding UTF-8

LazyData true

Collate 'Class-WaterLevelDataFrame.R' 'WaterLevelDataFrame-methods.R' 'data.R' 'getGaugingDataW.R' 'getPegelonlineCharacteristicValues.R' 'getPegelonlineW.R' 'hyd1d-internal.R' 'hyd1d.R' 'plotShiny.R' 'updateGaugingData.R' 'waterLevel.R' 'waterLevelFlood1.R' 'waterLevelFlood2.R' 'waterLevelFlys3.R' 'waterLevelFlys3InterpolateX.R' 'waterLevelFlys3InterpolateY.R' 'waterLevelFlys3Seq.R' 'waterLevelPegelonline.R' 'zzz.R'

VignetteBuilder knitr

BugReports https://gitlab.opencode.de/bafg-bund/hyd1d/-/work_items

URL https://hyd1d.bafg.de, https://gitlab.opencode.de/bafg-bund/hyd1d

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Arnd Weber [aut, cre] (ORCID: <https://orcid.org/0000-0002-5973-2770>),
Marcus Hatz [aut],
Wolfgang Stürmer [ctb],
Wilfried Wiechmann [ctb],
Benjamin Eberhardt [ctb]

Maintainer Arnd Weber <arnd.weber@bafg.de>

Repository CRAN

Date/Publication 2026-08-19 17:00:02 UTC

Contents

`as.data.frame.WaterLevelDataFrame` *Coerce a WaterLevelDataFrame to a data.frame*

Description

A function to coerce an object of class WaterLevelDataFrame to a `data.frame`.

Usage

```
## S3 method for class 'WaterLevelDataFrame'
as.data.frame(x, ...)
```

Arguments

`x`	an object of class WaterLevelDataFrame.
`...`	additional arguments to be passed to the internally used `as.data.frame`-function.

Value

`as.data.frame` returns a `data.frame`.

See Also

`WaterLevelDataFrame`, `data.frame`, `as.data.frame`

Examples

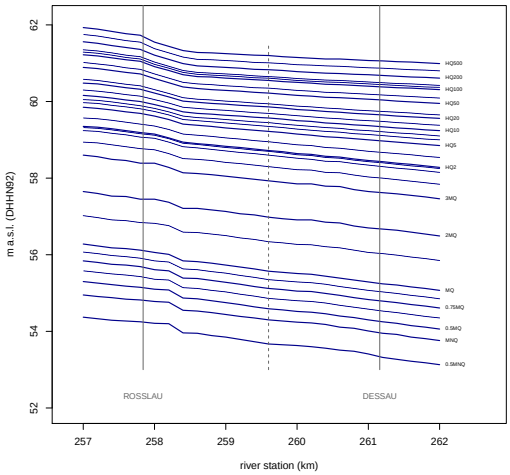
```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
df <- as.data.frame(wldf)
```

`df.flys` *Stationary water levels from the FLYS 3-database*

Description

This dataset contains the 30 stationary 1d water levels for the rivers **Elbe** and **Rhine** originally stored in the FLYS3-database.

For both rivers 30 stationary water levels have been computed by means of the 1d hydraulic model SOBEK. The water levels cover the full length of the free flowing river sections with a spatial resolution of 200 m river stretch along the official river stationing. They range from extremely low to extremely high flow conditions and are usually separated vertically by 0.2 - 0.6 m.



Usage

```
df.flys
```

Format

A `data.frame` with 169980 rows and 4 variables:

river name of the relevant water body (type `character`).

name of the FLYS 3 water level (type `character`). See details for more information.

station rivers stationing (type `numeric`).

w water level (cm above gauge zero, type `numeric`).

Details

The nameing of the water levels is river-specific:

Elbe:

'0.5MNQ', 'MNQ', '0.5MQ', 'a', '0.75MQ', 'b', 'MQ', 'c', '2MQ', '3MQ', 'd', 'e', 'MHQ', 'HQ2', 'f', 'HQ5', 'g', 'h', 'HQ10', 'HQ15', 'HQ20', 'HQ25', 'HQ50', 'HQ75', 'HQ100', 'i', 'HQ150', 'HQ200', 'HQ300', 'HQ500'

Rhine:

'Ud=1', 'Ud=5', 'GlQ2012', 'Ud=50', 'Ud=80', 'Ud=100', 'Ud=120', 'Ud=183', 'MQ', 'Ud=240', 'Ud=270', 'Ud=310', 'Ud=340', 'Ud=356', 'Ud=360', 'MHQ', 'HQ2', 'HQ5', 'HQ5-10', 'HQ10', 'HQ10-20', '~HQ20', 'HQ20-50', 'HQ50', 'HQ50-100', 'HQ100', 'HQ100-200', 'HQ200', 'HQ200-ex', 'HQextr.'

Both lists of water levels are ordered from low to high water levels.

References

Busch N, Hammer M (2009). "Einheitliche Grundlage für die Festlegung der Bemessungswasserspiegellagen der Elbe auf der frei fließenden Strecke in Deutschland." doi:10.5675/bfg1650.

HKV Hydrokontor (2014). "Erstellung eines SOBEK-River Modells für den Rhein von Iffezheim bis Pannerdense Kop als Weiterentwicklung bestehender SOBEK-RE Modelle."

Bundesanstalt für Gewässerkunde (2013). "FLYS goes WEB: Eröffnung eines neuen hydrologischen Fachdienstes in der BfG." doi:10.5675/BfG_Veranst_2013.4. https://doi.bafg.de/BfG/2013/Veranst4_2013.pdf.

Bundesanstalt für Gewässerkunde (2016). "FLYS – Flusshydrologischer Webdienst." https://www.bafg.de/DE/5_Informiert/1_Portale_Dienste/FLYS/flys_node.html.

DELTARES (2018). "SOBEK." https://download.deltares.nl/en/sobek/.

---

df.flys_sections *Reference gauging stations according to FLYS3*

Description

This dataset relates the reference gauging stations to river stationing as used within FLYS3

Usage

```
df.flys_sections
```

Format

A data.frame with 24 rows and 4 variables:

river name of the FLYS3 water body (type character).

gauging_station name of the reference gauging station (type character).

from uppermost station of the river section (type numeric).

to lowermost station of the river section (type numeric).

uuid name of the reference gauging station (type character).

References

Bundesanstalt für Gewässerkunde (2016). "FLYS – Flusshydrologischer Webdienst." https://www.bafg.de/DE/5_Informiert/1_Portale_Dienste/FLYS/flys_node.html.

`df.gauging_data` *Gauging data for all WSV-run gauging stations along Elbe and Rhine*

Description

This dataset contains all **daily-averaged** gauging data for the gauging stations along **Elbe** and **Rhine** operated by the waterway and shipping administration (Wasserstraßen- und Schifffahrtsverwaltung (WSV)) since 1960-01-01. Data from 1960-01-01 until 2025-12-31 are validated and were queried from (WISKI7)-database and supplied by <Datenstelle-M1@bafg.de>. Data after 2025-12-31 are continuously collected from https://pegelonline.wsv.de/gast/start and are not officially validated. Unvalidated recent data will be replaced anually and distributed through package and/or internal dataset updates.

The latest version is stored locally under `paste0(options()$hyd1d.datadir, "/df.gauging_data_latest.RDS")`. To modify the location of your locally stored gauging data set using `options()` prior to loading the package, e.g. `options("hyd1d.datadir" = "~/.hyd1d");library(hyd1d)`. The location can be determined through the environmental variable `hyd1d_datadir`.

Usage

```
df.gauging_data
```

Format

A `data.frame` with 1379334 (rows and 3 variables):

gauging_station name of the gauging station (type `character`). It is used as JOIN field for dataset `df.gauging_station_data`.

date of the measurement (type `Date`).

w water level relative to the gauge zero (cm, type `numeric`).

References

Wasserstraßen- und Schifffahrtsverwaltung des Bundes (WSV) (2026). "Pegeldaten für Elbe und Rhein."

Wasserstraßen- und Schifffahrtsverwaltung des Bundes (WSV) (2022). "PEGELONLINE." https://pegelonline.wsv.de/gast/start.

See Also

`updateGaugingData`

Examples

```
options("hyd1d.datadir" = tempdir())
updateGaugingData(paste0(options()$hyd1d.datadir,
                         "/df.gauging_data_latest.RDS"))
```

`df.gauging_station_data` *Gauging station data for all WSV-run gauging stations along Elbe and Rhine*

Description

This dataset contains gauging station data for the gauging stations along **Elbe** and **Rhine** operated by the waterway and shipping administration (Wasserstraßen- und Schifffahrtsverwaltung (WSV)). The data were originally obtained from https://pegelonline.wsv.de/gast/start and are updated anually.

Usage

```
df.gauging_station_data
```

Format

A `data.frame` with 70 rows and 13 variables:

id continuous numbering (type `integer`).

gauging_station name of the gauging station (type `character`). It is used as JOIN field for dataset `df.gauging_data`.

uuid of the gauging station in the PEGELONLINE system (type `character`).

agency of the waterway and shipping administration in charge of the respective gauging station (type `character`).

km official stationing of the gauging station (type `numeric`).

longitude of the gauging stations location (WGS1984, type `numeric`).

latitude of the gauging stations location (WGS1984, type `numeric`).

mw mean water level of the gauging station (m relative to the gauge zero, type `numeric`).

mw_timespan timespan used to derive the gauging stations mean water level (type `character`).

pnp the gauge zero relative to sea level (NHN (DHHN92), type `numeric`).

data_present `logical` to separate TRUE (real) from section structuring FALSE gauging stations.

km_qps corrected stationing used for the water level computations of `waterLevel` and `waterLevelPegelonline` (type `numeric`).

river the gauging station is located on (type `character`).

References

Wasserstraßen- und Schifffahrtsverwaltung des Bundes (WSV) (2022). "PEGELONLINE." https://pegelonline.wsv.de/gast/start.

getGaugingDataW *Get W from internal dataset df.gauging_data for the specified gauging station and time*

Description

Extract the daily mean water level data from df.gauging_data for specific gauging station and date.

Usage

```
getGaugingDataW(gauging_station, time, uuid)
```

Arguments

gauging_station

must be type character with a length of one. Permitted values are: 'SCHOENA', 'PIRNA', 'DRESDEN', 'MEISSEN', 'RIESA', 'MUEHLBERG', 'TORGAU', 'PRETZSCH-MAUKEN', 'ELSTER', 'WITTENBERG', 'COSWIG', 'VOCKERODE', 'ROSSLAU', 'DESSAU', 'AKEN', 'BARBY', 'SCHOENEBECK', 'MAGDEBURG-BUCKAU', 'MAGDEBURG-STROMBRUECKE', 'MAGDEBURG-ROTHENSEE', 'NIEGRIPP AP', 'ROGAETZ', 'TANGERMUENDE', 'STORKAU', 'SANDAU', 'SCHARLEUK', 'WITTENBERGE', 'MUEGGENDORF', 'SCHNACKENBURG', 'LENZEN', 'GORLEBEN', 'DOEMITZ', 'DAMNATZ', 'HITZACKER', 'NEU DARCHAU', 'BLECKEDE', 'BOIZENBURG', 'HOHNSTORF', 'ARTLENBURG', 'GEESTHACHT', 'IFFEZHEIM', 'PLITTERSDORF', 'MAXAU', 'PHILIPPSBURG', 'SPEYER', 'MANNHEIM', 'WORMS', 'NIERSTEIN-OPPENHEIM', 'MAINZ', 'OESTRICH', 'BINGEN', 'KAUB', 'SANKT GOAR', 'BOPPARD', 'BRAUBACH', 'KOBLENZ', 'ANDERNACH', 'OBERWINTER', 'BONN', 'KOELN', 'DUESSELDORF', 'RUHRORT', 'WESEL', 'REES', 'EMMERICH'.

time

must be type c("POSIXct", "POSIXlt") or Date and in the temporal range between 1960-01-01 and now (Sys.time() or Sys.Date()).

uuid

must be type character with a length of one. Permitted values are: '7cb7461b-3530-4c01-8978-7f676b8f71ed', '85d686f1-55b2-4d36-8dba-3207b50901a7', '70272185-b2b3-4178-96b8-43bea330dcae', '24440872-5bd2-4fb3-8554-907b49816c49', 'b04b739d-7ffa-41ee-9eb9-95cb1b4ef508', '16b9b4e7-be14-41fd-941e-6755c97276cc', '83bbaedb-5d81-4bc6-9f66-3bd700c99c1f', 'f3dc8f07-c2bb-4b92-b0b0-4e01a395a2c6', 'c093b557-4954-4f05-8f5c-6c6d7916c62d', '070b1eb4-3872-4e07-b2e5-e25fd9251b93', '1ce53a59-33b9-40dc-9b17-3cd2a2414607', 'ae93f2a5-612e-4514-b5fd-9c8aecdd73c7', 'e97116a4-7d30-4671-8ba1-cdce0a153d1d', '1edc5fa4-88af-47f5-95a4-0e77a06fe8b1', '094b96e5-caeb-46d3-a8ee-d44182add069', '939f82ec-15a9-49c8-8828-dc2f8a2d49e2', '90bcb315-f080-41a8-a0ac-6122331bb4cf', 'b8567c1e-8610-4c2b-a240-65e8a74919fa', 'ccccb57f-a2f9-4183-ae88-5710d3afaefd', 'e30f2e83-b80b-4b96-8f39-fa60317afcc7', '3adf88fd-fd7a-41d0-84f5-1143c98a6564', '133f0f6c-2ca1-4798-9360-5b5f417dd839', '13e91b77-90f3-41a5-a320-641748e9c311', 'de4cc1db-51cb-4b62-bee2-9750cbe4f5c4', 'f4c55f77-ab80-4e00-bed3-aa6631aba074', 'e32b0a28-8cd5-4053-bc86-fff9c6469106', 'cbf3cd49-91bd-49cc-8926-ccc6c0e7eca4', '48f2661f-f9cb-4093-9d57-da2418ed656e',

'550e3885-a9d1-4e55-bd25-34228bd6d988', 'c80a4f21-528c-4771-98d7-10cd591699a4', 'ac507f42-1593-49ea-865f-10b2523617c7', '6e3ea719-48b1-408a-bc55-0986c1e94cd5', 'c233674f-259a-4304-b81f-dce1f415d85b', 'a26e57c9-1cb8-4fca-ba80-9e02abc81df8', '67d6e882-b60c-40d3-975c-a6d7a2b4e40a', '6aa1cd8e-e528-4bcb-ba8e-705b6dcb7da2', '33e0bce0-13df-4ffc-be9d-f1a79e795e1c', 'd9289367-c8aa-4b6a-b1ad-857fec94c6bb', 'b3492c68-8373-4769-9b29-22f66635a478', '44f7e955-c97d-45c8-9ed7-19406806fb4c', 'b02be240-1364-4c97-8bb6-675d7d842332', '6b774802-fcb5-49ae-8ecb-ecaf1a278b1c', 'b6c6d5c8-e2d5-4469-8dd8-fa972ef7eaea', '88e972e1-88a0-4eb9-847c-0925e5999a46', '2cb8ae5b-c5c9-4fa8-bac0-bb724f2754f4', '57090802-c51a-4d09-8340-b4453cd0e1f5', '844a620f-f3b8-4b6b-8e3c-783ae2aa232a', 'd28e7ed1-3317-41c5-bec6-725369ed1171', 'a37a9aa3-45e9-4d90-9df6-109f3a28a5af', '665be0fe-5e38-43f6-8b04-02a93bdbeeb4', '0309cd61-90c9-470e-99d4-2ee4fb2c5f84', '1d26e504-7f9e-480a-b52c-5932be6549ab', '550eb7e9-172e-48e4-ae1e-d1b761b42223', '2ff6379d-d168-4022-8da0-16846d45ef9b', 'd6dc44d1-63ac-4871-b175-60ac4040069a', '4c7d796a-39f2-4f26-97a9-3aad01713e29', '5735892a-ec65-4b29-97c5-50939aa9584e', 'b45359df-c020-4314-adb1-d1921db642da', '593647aa-9fea-43ec-a7d6-6476a76ae868', 'a6ee8177-107b-47dd-bcfd-30960ccc6e9c', '8f7e5f92-1153-4f93-acba-ca48670c8ca9', 'c0f51e35-d0e8-4318-afaf-c5fcbc29f4c1', 'f33c3cc9-dc4b-4b77-baa9-5a5f10704398', '2f025389-fac8-4557-94d3-7d0428878c86', '9598e4cb-0849-401e-bba0-689234b27644'.

Details

This functions queries package-internal gauging data (`df.gauging_data`).

Value

If gauging data exist for the specified gauging station and time, a water level is returned. If no data exist, `NA` is returned.

References

Wasserstraßen- und Schifffahrtsverwaltung des Bundes (WSV) (2026). "Pegeldaten für Elbe und Rhein."

Examples

```
getGaugingDataW(gauging_station = "DESSAU", time = as.Date("2016-12-21"))
```

`getGaugingStations` *Extract a WaterLevelDataFrame's slot gauging_stations*

Description

A function to extract the slot `gauging_stations` from an object of class WaterLevelDataFrame.

Usage

```
getGaugingStations(x)

## S4 method for signature 'WaterLevelDataFrame'
getGaugingStations(x)
```

Arguments

x	an object of class WaterLevelDataFrame.

Value

The function above extracts the slot gauging_stations and returns an object of class data.frame, which might contain gauging station data that have been used for the interpolation of a water level for the specified date.

See Also

setGaugingStations<--method

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf <- waterLevel(wldf)
getGaugingStations(wldf)
```

---

getGaugingStationsMissing

Extract a WaterLevelDataFrame's slot gauging_stations_missing

---

Description

A function to extract the slot gauging_stations_missing from an object of class WaterLevelDataFrame.

Usage

```
getGaugingStationsMissing(x)

## S4 method for signature 'WaterLevelDataFrame'
getGaugingStationsMissing(x)
```

Arguments

x	an object of class WaterLevelDataFrame.

Value

The function above extracts the slot `gauging_stations_missing` and returns an object of class `character`, which might contain a vector with gauging stations without gauging data for the specified date.

See Also

`setGaugingStationsMissing<--method`

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("1991-12-16"),
                            station = seq(500, 501, 0.1))
wldf <- waterLevel(wldf)
getGaugingStationsMissing(wldf)
```

`getPegelonlineCharacteristicValues` *Query characteristic values from pegelonline.wsv.de for the specified gauging station*

Description

Download characteristic water level data for a gauging station from https://pegelonline.wsv.de/gast/start.

Usage

```
getPegelonlineCharacteristicValues(
  gauging_station,
  value,
  uuid,
  as_list = TRUE,
  abs_height = TRUE,
  verbose = TRUE
)
```

Arguments

`gauging_station` must be type `character` with a length of one. Permitted values are: 'SCHOENA', 'PIRNA', 'DRESDEN', 'MEISSEN', 'RIESA', 'MUEHLBERG', 'TORGAU', 'PRETZSCH-MAUKEN', 'ELSTER', 'WITTENBERG', 'COSWIG', 'VOCKERODE', 'ROSSLAU', 'DESSAU', 'AKEN', 'BARBY', 'SCHOENEBECK', 'MAGDEBURG-BUCKAU', 'MAGDEBURG-STROMBRUECKE', 'MAGDEBURG-ROTHENSEE',

'NIEGRIPP AP', 'ROGAETZ', 'TANGERMUENDE', 'STORKAU', 'SANDAU', 'SCHARLEUK', 'WITTENBERGE', 'MUEGGENDORF', 'SCHNACKENBURG', 'LENZEN', 'GORLEBEN', 'DOEMITZ', 'DAMNATZ', 'HITZACKER', 'NEU DARCHAU', 'BLECKEDE', 'BOIZENBURG', 'HOHNSTORF', 'ARTLENBURG', 'GEESTHACHT', 'WEHR GEESTHACHT UP', 'ALTENGAMME', 'ZOLLENSPIEKER', 'BUNTHAUS', 'HAMBURG ST. PAULI', 'SEEMANNSHOEFT', 'BLANKENESE UF', 'SCHULAU', 'LUEHORT', 'STADERSAND', 'KOLLMAR', 'GLUECKSTADT', 'BROKDORF', 'BRUNSBUETTEL MPM', 'OSTERIFF MPM', 'OTTERNDORF MPM', 'CUXHAVEN STEUBENHOEFT', 'MITTELGRUND', 'SCHARHOERN', 'BAKE Z', 'HERBRUM HAFENDAMM', 'RHEDE', 'PAPENBURG', 'WEENER', 'LEERORT', 'TERBORG', 'POGUM', 'EMDEN NEUE SEESCHLEUSE', 'KNOCK', 'DUKEGAT', 'EMSHOERN', 'BORKUM FISCHERBALJE', 'NORDERNEY RIFFGAT', 'LANGEOOG', 'SPIEKEROOG', 'IFFEZHEIM', 'PLITTERSDORF', 'MAXAU', 'PHILIPPSBURG', 'SPEYER', 'MANNHEIM', 'WORMS', 'NIERSTEIN-OPPENHEIM', 'MAINZ', 'OESTRICH', 'BINGEN', 'KAUB', 'SANKT GOAR', 'BOPPARD', 'BRAUBACH', 'KOBLENZ', 'ANDERNACH', 'OBERWINTER', 'BONN', 'KOELN', 'DUESSELDORF', 'RUHRORT', 'WESEL', 'REES', 'EMMERICH', 'GROENHUDE', 'BREITENBERG', 'ITZEHOE HAFEN', 'STOER-SPERRWERK BP'.

`value` must be type `character`. Commonly available values are `c("PNP", "MThw", "MTnw", "HThw", "NTnw", "HHW", "NNW", "MNW", "MW","MHW")`.

`uuid` must be type `character` with a length of one. Permitted values are: '7cb7461b-3530-4c01-8978-7f676b8f71ed', '85d686f1-55b2-4d36-8dba-3207b50901a7', '70272185-b2b3-4178-96b8-43bea330dcae', '24440872-5bd2-4fb3-8554-907b49816c49', 'b04b739d-7ffa-41ee-9eb9-95cb1b4ef508', '16b9b4e7-be14-41fd-941e-6755c97276cc', '83bbaedb-5d81-4bc6-9f66-3bd700c99c1f', 'f3dc8f07-c2bb-4b92-b0b0-4e01a395a2c6', 'c093b557-4954-4f05-8f5c-6c6d7916c62d', '070b1eb4-3872-4e07-b2e5-e25fd9251b93', '1ce53a59-33b9-40dc-9b17-3cd2a2414607', 'ae93f2a5-612e-4514-b5fd-9c8aecdd73c7', 'e97116a4-7d30-4671-8ba1-cdce0a153d1d', '1edc5fa4-88af-47f5-95a4-0e77a06fe8b1', '094b96e5-caeb-46d3-a8ee-d44182add069', '939f82ec-15a9-49c8-8828-dc2f8a2d49e2', '90bcb315-f080-41a8-a0ac-6122331bb4cf', 'b8567c1e-8610-4c2b-a240-65e8a74919fa', 'ccccb57f-a2f9-4183-ae88-5710d3afaefd', 'e30f2e83-b80b-4b96-8f39-fa60317afcc7', '3adf88fd-fd7a-41d0-84f5-1143c98a6564', '133f0f6c-2ca1-4798-9360-5b5f417dd839', '13e91b77-90f3-41a5-a320-641748e9c311', 'de4cc1db-51cb-4b62-bee2-9750cbe4f5c4', 'f4c55f77-ab80-4e00-bed3-aa6631aba074', 'e32b0a28-8cd5-4053-bc86-fff9c6469106', 'cbf3cd49-91bd-49cc-8926-ccc6c0e7eca4', '48f2661f-f9cb-4093-9d57-da2418ed656e', '550e3885-a9d1-4e55-bd25-34228bd6d988', 'c80a4f21-528c-4771-98d7-10cd591699a4', 'ac507f42-1593-49ea-865f-10b2523617c7', '6e3ea719-48b1-408a-bc55-0986c1e94cd5', 'c233674f-259a-4304-b81f-dce1f415d85b', 'a26e57c9-1cb8-4fca-ba80-9e02abc81df8', '67d6e882-b60c-40d3-975c-a6d7a2b4e40a', '6aa1cd8e-e528-4bcb-ba8e-705b6dcb7da2', '33e0bce0-13df-4ffc-be9d-f1a79e795e1c', 'd9289367-c8aa-4b6a-b1ad-857fec94c6bb', 'b3492c68-8373-4769-9b29-22f66635a478', '44f7e955-c97d-45c8-9ed7-19406806fb4c', '0f7f58a8-411f-43d9-b42a-e897e63c4faa', '2ee12b9a-f7fd-4856-82b9-6bdd850c2bba', '3de8ea26-ab29-4e46-adad-06198ba2e0b7', 'ae1b91d0-e746-4f65-9f64-2d2e23603a82', 'd488c5cc-4de9-4631-8ce1-0db0e700b546', '816affba-0118-4668-887f-fb882ed573b2', 'bacb459b-0f24-4233-bb35-cd224a51678e', 'f3c6ee73-5561-4068-96ec-364016e7d9ef', '8d18d129-07f1-4c4d-adba-a985016be0b0', '80f0fc4d-9fc7-449d-9d68-ee89333f0eff', '3ed90357-4b01-4119-b1c5-bd2c62871e7b', '1f1bbed7-c1fa-45b4-90d3-df94b50ad631',

'610ab204-d3c4-4a11-a38b-e31461fdcf27', 'd4f5f719-8c52-4f8d-945d-1c31404cc628', 'eb90bd3f-5405-412d-81e0-7a58be52dcef', '5140295e-b93e-4081-a920-642d89c7ca8b', 'aad49293-242a-43ad-a8b1-e91d7792c4b2', '3ff99b92-4396-4fa7-af73-02b9c015dcad', 'f0197bcf-6846-4c0a-9659-0c2626a9bcf0', '104fdc24-1dc6-4cb7-b44f-10bd02e13f40', '8177a148-5674-4b8f-8ded-050907f640f3', '16508b11-4349-48f7-be51-1227b7888585', 'ec4a598d-773d-44c1-935e-2053b54e45a3', 'aa6af4e6-a44f-46c4-abf6-449f8a68bab1', 'abb23dad-0880-41ab-8d2d-dd33e11f148f', '244cae8b-ce75-4c2d-a66e-cb804f8335a2', '5d1e4350-0f39-4428-84c3-6f8f0bbe80d4', 'edfdf747-be92-462f-87ed-53d228a33172', '438b565e-f293-43c8-8771-377e555ed5ec', '7753c1fa-34d8-4d09-a7c7-38024079117c', 'c8af067c-ba6a-4a76-86d8-1ce8e532ef8b', '8727ebfd-e2e1-43da-ab3d-fee48cff9acc', 'c0244c0e-6ae6-40cb-a967-4039b2a0ce7c', 'a0c1dcb6-7812-48e6-8c01-f7edad7a2caf', '662c4b5e-0241-456d-ac7d-9f62fd95c0d1', 'b02be240-1364-4c97-8bb6-675d7d842332', '6b774802-fcb5-49ae-8ecb-ecaf1a278b1c', 'b6c6d5c8-e2d5-4469-8dd8-fa972ef7eaea', '88e972e1-88a0-4eb9-847c-0925e5999a46', '2cb8ae5b-c5c9-4fa8-bac0-bb724f2754f4', '57090802-c51a-4d09-8340-b4453cd0e1f5', '844a620f-f3b8-4b6b-8e3c-783ae2aa232a', 'd28e7ed1-3317-41c5-bec6-725369ed1171', 'a37a9aa3-45e9-4d90-9df6-109f3a28a5af', '665be0fe-5e38-43f6-8b04-02a93bdbeeb4', '0309cd61-90c9-470e-99d4-2ee4fb2c5f84', '1d26e504-7f9e-480a-b52c-5932be6549ab', '550eb7e9-172e-48e4-ae1e-d1b761b42223', '2ff6379d-d168-4022-8da0-16846d45ef9b', 'd6dc44d1-63ac-4871-b175-60ac4040069a', '4c7d796a-39f2-4f26-97a9-3aad01713e29', '5735892a-ec65-4b29-97c5-50939aa9584e', 'b45359df-c020-4314-adb1-d1921db642da', '593647aa-9fea-43ec-a7d6-6476a76ae868', 'a6ee8177-107b-47dd-bcfd-30960ccc6e9c', '8f7e5f92-1153-4f93-acba-ca48670c8ca9', 'c0f51e35-d0e8-4318-afaf-c5fcbc29f4c1', 'f33c3cc9-dc4b-4b77-baa9-5a5f10704398', '2f025389-fac8-4557-94d3-7d0428878c86', '9598e4cb-0849-401e-bba0-689234b27644', '15859426-834c-429e-9c41-2e097b717b1d', '24c6a014-864b-4d53-bd05-0b49106f5412', 'd863cbc3-5e5e-4095-855c-026f0850dd58', 'e5b8e9f3-f0cc-4ad7-8707-577ee1b25b3e'.

`as_list` 'boolean' to switch between 'list' or 'data.frame' output

`abs_height` 'boolean' to switch between absolute and relative height output

`verbose` 'boolean' to switch off warnings

Details

This functions queries online data through the REST service of PEGELONLINE.

Value

The returned output is a named `list` or `data.frame` with the queried characteristic value(s) and potentially other relevant information such as time spans.

References

Wasserstraßen- und Schifffahrtsverwaltung des Bundes (WSV) (2022). "PEGELONLINE." https://pegelonline.wsv.de/gast/start.

Examples

```
getPegelonlineCharacteristicValues(gauging_station = "DESSAU", value = "MW",
                                   as_list = FALSE)
getPegelonlineCharacteristicValues(gauging_station = "HAMBURG ST. PAULI",
```

```
value = "MThw")
```

`getPegelonlineW` *Get W from pegelonline.wsv.de for the specified gauging station and time*

Description

Download and temporarily interpolate or average water level data from https://pegelonline.wsv.de/gast/start.

Usage

```
getPegelonlineW(gauging_station, time, uuid)
```

Arguments

`gauging_station`

must be type `character` with a length of one. Permitted values are: 'SCHOENA', 'PIRNA', 'DRESDEN', 'MEISSEN', 'RIESA', 'MUEHLBERG', 'TORGAU', 'PRETZSCH-MAUKEN', 'ELSTER', 'WITTENBERG', 'COSWIG', 'VOCKERODE', 'ROSSLAU', 'DESSAU', 'AKEN', 'BARBY', 'SCHOENEBECK', 'MAGDEBURG-BUCKAU', 'MAGDEBURG-STROMBRUECKE', 'MAGDEBURG-ROTHENSEE', 'NIEGRIPP AP', 'ROGAETZ', 'TANGERMUENDE', 'STORKAU', 'SANDAU', 'SCHARLEUK', 'WITTENBERGE', 'MUEGGENDORF', 'SCHNACKENBURG', 'LENZEN', 'GORLEBEN', 'DOEMITZ', 'DAMNATZ', 'HITZACKER', 'NEU DARCHAU', 'BLECKEDE', 'BOIZENBURG', 'HOHNSTORF', 'ARTLENBURG', 'GEESTHACHT', 'WEHR GEESTHACHT UP', 'ALTENGAMME', 'ZOLLENSPIEKER', 'BUNTHAUS', 'HAMBURG ST. PAULI', 'SEEMANNSHOEFT', 'BLANKENESE UF', 'SCHULAU', 'LUEHORT', 'STADERSAND', 'KOLLMAR', 'GLUECKSTADT', 'BROKDORF', 'BRUNSBUETTEL MPM', 'OSTERIFF MPM', 'OTTERNDORF MPM', 'CUXHAVEN STEUBENHOEFT', 'MITTELGRUND', 'SCHARHOERN', 'BAKE Z', 'HERBRUM HAFENDAMM', 'RHEDE', 'PAPENBURG', 'WEENER', 'LEERORT', 'TERBORG', 'POGUM', 'EMDEN NEUE SEESCHLEUSE', 'KNOCK', 'DUKEGAT', 'EMSHOERN', 'BORKUM FISCHERBALJE', 'NORDERNEY RIFFGAT', 'LANGEOOG', 'SPIEKEROOG', 'IFFEZHEIM', 'PLITTERSDORF', 'MAXAU', 'PHILIPPSBURG', 'SPEYER', 'MANNHEIM', 'WORMS', 'NIERSTEIN-OPPENHEIM', 'MAINZ', 'OESTRICH', 'BINGEN', 'KAUB', 'SANKT GOAR', 'BOPPARD', 'BRAUBACH', 'KOBLENZ', 'ANDERNACH', 'OBERWINTER', 'BONN', 'KOELN', 'DUESSELDORF', 'RUHRORT', 'WESEL', 'REES', 'EMMERICH', 'GROENHUDE', 'BREITENBERG', 'ITZEHOE HAFEN', 'STOER-SPERRWERK BP'.

`time`

must be type `c("POSIXct", "POSIXt")` or `Date` and be in the temporal range between 31 days ago `Sys.time() - 2678400` or `Sys.Date() - 31` and now (`Sys.time()`) or yesterday (`Sys.Date() - 1`).

uuid must be type character with a length of one. Permitted values are: '7cb7461b-3530-4c01-8978-7f676b8f71ed', '85d686f1-55b2-4d36-8dba-3207b50901a7', '70272185-b2b3-4178-96b8-43bea330dcae', '24440872-5bd2-4fb3-8554-907b49816c49', 'b04b739d-7ffa-41ee-9eb9-95cb1b4ef508', '16b9b4e7-be14-41fd-941e-6755c97276cc', '83bbaedb-5d81-4bc6-9f66-3bd700c99c1f', 'f3dc8f07-c2bb-4b92-b0b0-4e01a395a2c6', 'c093b557-4954-4f05-8f5c-6c6d7916c62d', '070b1eb4-3872-4e07-b2e5-e25fd9251b93', '1ce53a59-33b9-40dc-9b17-3cd2a2414607', 'ae93f2a5-612e-4514-b5fd-9c8aecdd73c7', 'e97116a4-7d30-4671-8ba1-cdce0a153d1d', '1edc5fa4-88af-47f5-95a4-0e77a06fe8b1', '094b96e5-caeb-46d3-a8ee-d44182add069', '939f82ec-15a9-49c8-8828-dc2f8a2d49e2', '90bcb315-f080-41a8-a0ac-6122331bb4cf', 'b8567c1e-8610-4c2b-a240-65e8a74919fa', 'ccccb57f-a2f9-4183-ae88-5710d3afaefd', 'e30f2e83-b80b-4b96-8f39-fa60317afcc7', '3adf88fd-fd7a-41d0-84f5-1143c98a6564', '133f0f6c-2ca1-4798-9360-5b5f417dd839', '13e91b77-90f3-41a5-a320-641748e9c311', 'de4cc1db-51cb-4b62-bee2-9750cbe4f5c4', 'f4c55f77-ab80-4e00-bed3-aa6631aba074', 'e32b0a28-8cd5-4053-bc86-fff9c6469106', 'cbf3cd49-91bd-49cc-8926-ccc6c0e7eca4', '48f2661f-f9cb-4093-9d57-da2418ed656e', '550e3885-a9d1-4e55-bd25-34228bd6d988', 'c80a4f21-528c-4771-98d7-10cd591699a4', 'ac507f42-1593-49ea-865f-10b2523617c7', '6e3ea719-48b1-408a-bc55-0986c1e94cd5', 'c233674f-259a-4304-b81f-dce1f415d85b', 'a26e57c9-1cb8-4fca-ba80-9e02abc81df8', '67d6e882-b60c-40d3-975c-a6d7a2b4e40a', '6aa1cd8e-e528-4bcb-ba8e-705b6dcb7da2', '33e0bce0-13df-4ffc-be9d-f1a79e795e1c', 'd9289367-c8aa-4b6a-b1ad-857fec94c6bb', 'b3492c68-8373-4769-9b29-22f66635a478', '44f7e955-c97d-45c8-9ed7-19406806fb4c', '0f7f58a8-411f-43d9-b42a-e897e63c4faa', '2ee12b9a-f7fd-4856-82b9-6bdd850c2bba', '3de8ea26-ab29-4e46-adad-06198ba2e0b7', 'ae1b91d0-e746-4f65-9f64-2d2e23603a82', 'd488c5cc-4de9-4631-8ce1-0db0e700b546', '816affba-0118-4668-887f-fb882ed573b2', 'bacb459b-0f24-4233-bb35-cd224a51678e', 'f3c6ee73-5561-4068-96ec-364016e7d9ef', '8d18d129-07f1-4c4d-adba-a985016be0b0', '80f0fc4d-9fc7-449d-9d68-ee89333f0eff', '3ed90357-4b01-4119-b1c5-bd2c62871e7b', '1f1bbed7-c1fa-45b4-90d3-df94b50ad631', '610ab204-d3c4-4a11-a38b-e31461fdcf27', 'd4f5f719-8c52-4f8d-945d-1c31404cc628', 'eb90bd3f-5405-412d-81e0-7a58be52dcef', '5140295e-b93e-4081-a920-642d89c7ca8b', 'aad49293-242a-43ad-a8b1-e91d7792c4b2', '3ff99b92-4396-4fa7-af73-02b9c015dcad', 'f0197bcf-6846-4c0a-9659-0c2626a9bcf0', '104fdc24-1dc6-4cb7-b44f-10bd02e13f40', '8177a148-5674-4b8f-8ded-050907f640f3', '16508b11-4349-48f7-be51-1227b7888585', 'ec4a598d-773d-44c1-935e-2053b54e45a3', 'aa6af4e6-a44f-46c4-abf6-449f8a68bab1', 'abb23dad-0880-41ab-8d2d-dd33e11f148f', '244cae8b-ce75-4c2d-a66e-cb804f8335a2', '5d1e4350-0f39-4428-84c3-6f8f0bbe80d4', 'edfdf747-be92-462f-87ed-53d228a33172', '438b565e-f293-43c8-8771-377e555ed5ec', '7753c1fa-34d8-4d09-a7c7-38024079117c', 'c8af067c-ba6a-4a76-86d8-1ce8e532ef8b', '8727ebfd-e2e1-43da-ab3d-fee48cff9acc', 'c0244c0e-6ae6-40cb-a967-4039b2a0ce7c', 'a0c1dcb6-7812-48e6-8c01-f7edad7a2caf', '662c4b5e-0241-456d-ac7d-9f62fd95c0d1', 'b02be240-1364-4c97-8bb6-675d7d842332', '6b774802-fcb5-49ae-8ecb-ecaf1a278b1c', 'b6c6d5c8-e2d5-4469-8dd8-fa972ef7eaea', '88e972e1-88a0-4eb9-847c-0925e5999a46', '2cb8ae5b-c5c9-4fa8-bac0-bb724f2754f4', '57090802-c51a-4d09-8340-b4453cd0e1f5', '844a620f-f3b8-4b6b-8e3c-783ae2aa232a', 'd28e7ed1-3317-41c5-bec6-725369ed1171', 'a37a9aa3-45e9-4d90-9df6-109f3a28a5af', '665be0fe-5e38-43f6-8b04-02a93bdbeeb4', '0309cd61-90c9-470e-99d4-2ee4fb2c5f84', '1d26e504-7f9e-480a-b52c-5932be6549ab', '550eb7e9-172e-48e4-ae1e-d1b761b42223', '2ff6379d-d168-4022-8da0-16846d45ef9b', 'd6dc44d1-63ac-4871-b175-60ac4040069a', '4c7d796a-39f2-4f26-97a9-3aad01713e29', '5735892a-ec65-4b29-97c5-50939aa9584e', 'b45359df-c020-4314-adb1-d1921db642da', '593647aa-9fea-43ec-a7d6-6476a76ae868',

'a6ee8177-107b-47dd-bcfd-30960ccc6e9c', '8f7e5f92-1153-4f93-acba-ca48670c8ca9', 'c0f51e35-d0e8-4318-afaf-c5fcbc29f4c1', 'f33c3cc9-dc4b-4b77-baa9-5a5f10704398', '2f025389-fac8-4557-94d3-7d0428878c86', '9598e4cb-0849-401e-bba0-689234b27644', '15859426-834c-429e-9c41-2e097b717b1d', '24c6a014-864b-4d53-bd05-0b49106f5412', 'd863cbc3-5e5e-4095-855c-026f0850dd58', 'e5b8e9f3-f0cc-4ad7-8707-577ee1b25b3e'.

Details

This functions queries online water level data through the REST service of PEGELONLINE. The gauging data from PEGELONLINE have a high temporal resolution of 15 minutes, enabling meaningful linear temporal interpolation if `time` is supplied with type `c("POSIXct", "POSIXt")`. If `time` is supplied with type `Date` water level data are aggregated to daily averages.

Since data from PEGELONLINE expire after 31 days, this function is only applicable to query unvalidated water level values for the last 31 days before function call. If you need older and validated data, feel free to contact the data service at the Federal Institute of Hydrology by email (<Datenstelle-M1@bafg.de>).

Value

The returned output depends on the type of the input parameter `time`. If `time` is type `c("POSIXct", "POSIXt")` the returned object contains queried and interpolated water levels. If `time` is type `Date` the returned object contains daily averaged water levels.

References

Wasserstraßen- und Schifffahrtsverwaltung des Bundes (WSV) (2022). "PEGELONLINE." https://pegelonline.wsv.de/gast/start.

See Also

`waterLevelPegelonline`

Examples

```
getPegelonlineW(gauging_station = "DESSAU", time = Sys.time() - 3600)
getPegelonlineW(gauging_station = "DESSAU", time = Sys.Date() - 1)
```

`getRiver` *Extract a WaterLevelDataFrame's slot river*

Description

A function to extract the slot `river` from an object of class WaterLevelDataFrame.

Usage

```
getRiver(x)

## S4 method for signature 'WaterLevelDataFrame'
getRiver(x)
```

Arguments

x	an object of class WaterLevelDataFrame.

Value

The function above extracts the slot `river` and returns an object of class `character`.

See Also

`setRiver<--method`

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
getRiver(wldf)
```

---

getTime *Extract a WaterLevelDataFrame's slot time*

Description

A function to extract the slot `time` from an object of class WaterLevelDataFrame.

Usage

```
getTime(x)

## S4 method for signature 'WaterLevelDataFrame'
getTime(x)
```

Arguments

x	an object of class WaterLevelDataFrame.

Value

The function above extracts the slot `time` and returns an object of type `c("POSIXct", "POSIXt")`.

See Also

setTime<--method

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
getTime(wldf)
```

---

hyd1d *hyd1d - 1d Water Level Interpolation along the Rivers Elbe and Rhine*

Description

The hyd1d package provides an S4 class, relevant datasets and functions to compute 1d water levels along the German federal waterways Elbe and Rhine.

S4 class WaterLevelDataFrame

The detailled description of the S4 class WaterLevelDataFrame is available here. This class structures the handling and computation of the 1d water levels.

Datasets

Datasets delivered with this package are:

- df.gauging_data
- df.gauging_station_data
- df.flys
- df.flys_sections

Water level computation

Water levels are either obtained from the df.flys-dataset by the functions waterLevelFlys3 or waterLevelFlys3Seq or computed by the functions waterLevel and waterLevelPegelonline. The later functions use the datasets df.flys and df.gauging_station_data and gauging data provided by df.gauging_data or https://pegelonline.wsv.de/gast/start to linearily interpolate continuous water levels intersecting with the measured water level data at the gauging stations.

Author(s)

Maintainer: Arnd Weber <arnd.weber@bafg.de> (ORCID)

Authors:

- Arnd Weber <arnd.weber@bafg.de> (ORCID)
- Marcus Hatz <hatz@bafg.de>

Other contributors:

- Wolfgang Stürmer [contributor]
- Wilfried Wiechmann <wiechmann@bafg.de> [contributor]
- Benjamin Eberhardt <eberhardt@bafg.de> [contributor]

See Also

Useful links:

- https://hyd1d.bafg.de
- https://gitlab.opencode.de/bafg-bund/hyd1d
- Report bugs at https://gitlab.opencode.de/bafg-bund/hyd1d/-/work_items

---

names<-,WaterLevelDataFrame,character-method

Set names of a WaterLevelDataFrame

Description

Function to get or set the column names of an object of class WaterLevelDataFrame.

Usage

```
## S4 replacement method for signature 'WaterLevelDataFrame,character'
names(x) <- value
```

Arguments

x	an object of class WaterLevelDataFrame.
value	a character vector of up to the same length as ncol(x). Since the names of the first three columns of an object of class WaterLevelDataFrame are predetermined ("station","station_int", "w") only the later names of additional columns can be modified.

Value

For names, a character vector of the same length as ncol(x).

For names<-, the updated object. (Note that the value of names(x) <- value is that of the assignment, value, not the return value from the left-hand side.)

Note

To access the slot names of an object of class WaterLevelDataFrame the function slotNames has to be used.

See Also

names, slotNames

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf <- waterLevel(wldf, TRUE)
names(wldf) <- c(names(wldf)[1:5], "WEIGHT_Y")
```

---

plotShiny *Plot a WaterLevelDataFrame in Shiny*

Description

This convenience function enables the easy visualisation of interpolated water levels stored as WaterLevelDataFrame using the R package shiny. The results of functions like waterLevel and waterLevelPegelonline can be plotted interactively so that the computation process itself becomes visible.

Usage

```
plotShiny(
  wldf,
  add_flys = TRUE,
  add_flys_labels = TRUE,
  add_weighting = TRUE,
  ...
)
```

Arguments

wldf	an object of class WaterLevelDataFrame.
add_flys	logical determining whether the used FLYS3 water levels should be plotted.
add_flys_labels	logical determining whether the used FLYS3 water levels should be labelled.
add_weighting	logical determining whether the weighting of gauging data at the gauging stations should be labelled.
...	further graphical parameters passed to plot.default.

Value

A plot of a WaterLevelDataFrame.

References

Bundesanstalt für Gewässerkunde (2016). "FLYS – Flusshydrologischer Webdienst." https://www.bafg.de/DE/5_Informiert/1_Portale_Dienste/FLYS/flys_node.html.

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf <- waterLevel(wldf, shiny = TRUE)
plotShiny(wldf, TRUE, TRUE, TRUE)
```

`rbind.WaterLevelDataFrame`

Combine WaterLevelDataFrames by Rows

Description

Take WaterLevelDataFrames that were produced for the same `river` and `time` and combine them by rows.

Usage

```
## S3 method for class 'WaterLevelDataFrame'
rbind(...)
```

Arguments

`...`	objects of class WaterLevelDataFrame.

Value

All supplied objects of class WaterLevelDataFrame will be combined to one object of class WaterLevelDataFrame which is returned.

See Also

`rbind`

Examples

```
wldf1 <- WaterLevelDataFrame(river   = "Elbe",
                             time    = as.POSIXct("2016-12-21"),
                             station = seq(257, 262, 0.1))
wldf2 <- WaterLevelDataFrame(river = "Elbe",
                             time = as.POSIXct("2016-12-21"),
                             station = seq(262, 270, 0.1))
wldf <- rbind(wldf1, wldf2)
```

setGaugingStations<- *Set a WaterLevelDataFrame's slot gauging_stations*

Description

A function to set the slot gauging_stations of an object of class WaterLevelDataFrame.

Usage

```
setGaugingStations(x) <- value

## S4 replacement method for signature 'WaterLevelDataFrame,data.frame'
setGaugingStations(x) <- value
```

Arguments

x	an object of class WaterLevelDataFrame.
value	a new value of class data.frame for the gauging_stations slot. value has to be a data.frame with the following columns and column types: id (integer), gauging_station (character), uuid (character), km (numeric), km_qps (numeric), water_shortname (character), longitude (numeric), latitude (numeric), mw (numeric), pnp (numeric), w (numeric), wl (numeric), n_wls_below_w_do (integer), n_wls_above_w_do (integer), n_wls_below_w_up (integer), n_wls_above_w_up (integer), name_wl_below_w_do (character), name_wl_above_w_do (character), name_wl_below_w_up (character), name_wl_above_w_up (character), w_wl_below_w_do (numeric), w_wl_above_w_do (numeric), w_wl_below_w_up (numeric), w_wl_above_w_up (numeric), weight_up (numeric), weight_do (numeric).

Value

The function sets a new value for the slot gauging_stations and returns an object of class WaterLevelDataFrame. Since value is normally generated inside the functions waterLevel or waterLevelPegelonline this function is of very little use outside these functions.

See Also

getGaugingStations-method

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf <- waterLevel(wldf)

df <- data.frame(id = integer(),
                 gauging_station    = character(),
                 uuid               = character(),
```

```
                  km                = numeric(),
                  km_qps            = numeric(),
                  river             = character(),
                  longitude         = numeric(),
                  latitude          = numeric(),
                  mw                = numeric(),
                  mw_timespan       = character(),
                  pnp               = numeric(),
                  w                 = numeric(),
                  wl                = numeric(),
                  n_wls_below_w_do  = integer(),
                  n_wls_above_w_do  = integer(),
                  n_wls_below_w_up  = integer(),
                  n_wls_above_w_up  = integer(),
                  name_wl_below_w_do = character(),
                  name_wl_above_w_do = character(),
                  name_wl_below_w_up = character(),
                  name_wl_above_w_up = character(),
                  w_wl_below_w_do   = numeric(),
                  w_wl_above_w_do   = numeric(),
                  w_wl_below_w_up   = numeric(),
                  w_wl_above_w_up   = numeric(),
                  weight_up         = numeric(),
                  weight_do         = numeric(),
                  stringsAsFactors  = FALSE)
setGaugingStations(wldf) <- df
```

`setGaugingStationsMissing<-` *Set a WaterLevelDataFrame's slot gauging_stations_missing*

Description

A function to set the slot `gauging_stations_missing` of an object of class WaterLevelDataFrame.

Usage

```
setGaugingStationsMissing(x) <- value

## S4 replacement method for signature 'WaterLevelDataFrame,character'
setGaugingStationsMissing(x) <- value
```

Arguments

`x`	an object of class WaterLevelDataFrame.
`value`	a new value of class `character` for the `gauging_stations_missing` slot.

Value

The function above sets a new value for the slot gauging_stations_missing and returns an object of class WaterLevelDataFrame.

See Also

getGaugingStationsMissing-method

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
setGaugingStationsMissing(wldf) <- as.character("VOCKERODE")
```

---

setRiver<- *Set a WaterLevelDataFrame's slot river*

Description

A function to set the slot river of an object of class WaterLevelDataFrame.

Usage

```
setRiver(x) <- value

## S4 replacement method for signature 'WaterLevelDataFrame,character'
setRiver(x) <- value
```

Arguments

x	an object of class WaterLevelDataFrame.
value	a new value of class character for the river slot. value has to have a length of one and has to be **Elbe** or **Rhine**.

Value

The function above sets a new value for the slot river and returns an object of class WaterLevelDataFrame. Since river is a slot relevant for the computation of the data.frame column w, w is set to NA and needs to be recomputed by functions like waterLevel or waterLevelPegelonline.

See Also

getRiver-method

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(500, 501, 0.1))
setRiver(wldf) <- as.character("Rhine")
```

---

setTime<- *Set a WaterLevelDataFrame's slot time*

Description

A function to set the slot `time` of an object of class WaterLevelDataFrame.

Usage

```
setTime(x) <- value

## S4 replacement method for signature 'WaterLevelDataFrame,POSIXct'
setTime(x) <- value

## S4 replacement method for signature 'WaterLevelDataFrame,POSIXlt'
setTime(x) <- value

## S4 replacement method for signature 'WaterLevelDataFrame,Date'
setTime(x) <- value
```

Arguments

x	an object of class WaterLevelDataFrame.
value	a new value of class `c("POSIXct", "POSIXt")` for the `time` slot. `value` has to have a length of one and has to be in the temporal range between `1960-01-01 00:00:00 CET` and now (`Sys.time()` or `NA`.

Value

The function above sets a new `value` for the slot `time` and returns an object of class WaterLevelDataFrame. Since `time` is a slot relevant for the computation of the `data.frame` column `w`, `w` is set to `NA` and needs to be recomputed by functions like `waterLevel` or `waterLevelPegelonline`.

See Also

`getTime-method`

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
setTime(wldf) <- as.POSIXct("2016-12-22")
```

`subset.WaterLevelDataFrame`

Subsetting WaterLevelDataFrames

Description

Returns subsets of WaterLevelDataFrames which meet conditions.

Usage

```
## S3 method for class 'WaterLevelDataFrame'
subset(x, subset, select, drop = FALSE, ...)
```

Arguments

`x`	object of class WaterLevelDataFrame.
`subset`	logical expression indicating elements or rows to keep: missing values are taken as false.
`select`	expression, indicating columns to select from a data frame.
`drop`	passed on to [indexing operator.
`...`	further arguments to be passed to or from other methods.

Value

An object similar to x, containing just the selected rows and columns. All other slots of the WaterLevelDataFrame remain unchanged.

See Also

`subset`

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf <- subset(wldf, station >= 258 & station <= 261)
```

summary.WaterLevelDataFrame *WaterLevelDataFrame summary*

Description

Returns a list of descriptive statistics for an object of class WaterLevelDataFrame.

Usage

```
## S3 method for class 'WaterLevelDataFrame'
summary(object, ...)
```

Arguments

object	an object of class WaterLevelDataFrame for which a summary is desired.
...	additional arguments to be passed to internally used functions.

Value

A list of summary statistics of the WaterLevelDataFrame and its slots.

See Also

summary

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf <- waterLevel(wldf)
summary(wldf)
```

updateGaugingData *Update local copy of* df.gauging data

Description

Function to overwrite and update the internal dataset df.gauging_data. This function is usually called during the initial loading of the package. If an update of df.gauging_data took place more than 8 days ago, an updated version of df.gauging_data will be downloaded and used.

Usage

```
updateGaugingData(x)
```

Arguments

x	path to the file containing df.gauging_data (type character).

Value

invisible(logical) notifying whether an updated version of df.gauging_data has been downloaded.

Examples

```
options("hyd1d.datadir" = tempdir())
updateGaugingData(paste0(options()$hyd1d.datadir,
                         "/df.gauging_data_latest.RDS"))
```

---

waterLevel *Compute a 1d water level dataset*

Description

Functions to compute 1d water level information and store it as column w of an S4 object of type WaterLevelDataFrame.

Usage

```
waterLevel(wldf, shiny = FALSE)

waterLevelPegelonline(wldf, shiny = FALSE)
```

Arguments

wldf	an object of class WaterLevelDataFrame.
shiny	logical determing whether columns (section, weight_x, weight_y) relevant for the plotShiny()-function are appended to the resulting WaterLevelDataFrame.

Details

waterLevel interpolates 1d water level along the river axis of Elbe and Rhine based on daily averaged, mostly validated gauging data stored in the internal dataset df.gauging_data. Internally stored gauging data are available from 1960-01-01 until yesterday.

waterLevelPegelonline carries out the interpolation with gauging data obtained through a REST service from https://pegelonline.wsv.de/gast/start. The gauging data from PEGELONLINE have a high temporal resolution of 15 minutes, enabling meaningful linear temporal interpolation. Since data from PEGELONLINE expire after 31 days, this function is only applicable for WaterLevelDataFrames with a time-slot set to appropriate values within the last 31 days before function call.

Value

An object of class WaterLevelDataFrame.

References

Busch N, Hammer M (2009). "Einheitliche Grundlage für die Festlegung der Bemessungswasserspiegellagen der Elbe auf der frei fließenden Strecke in Deutschland." doi:10.5675/bfg1650.

HKV Hydrokontor (2014). "Erstellung eines SOBEK-River Modells für den Rhein von Iffezheim bis Pannerdense Kop als Weiterentwicklung bestehender SOBEK-RE Modelle."

Bundesanstalt für Gewässerkunde (2016). "FLYS – Flusshydrologischer Webdienst." https://www.bafg.de/DE/5_Informiert/1_Portale_Dienste/FLYS/flys_node.html.

Wasserstraßen- und Schifffahrtsverwaltung des Bundes (WSV) (2022). "PEGELONLINE." https://pegelonline.wsv.de/gast/start.

See Also

plotShiny

Examples

```
# waterLevel
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf <- waterLevel(wldf)

# waterLevelPegelonline
wldf1 <- wldf
setTime(wldf1) <- Sys.time() - as.difftime(60, units = "mins")
wldf1 <- waterLevelPegelonline(wldf1)
```

WaterLevelDataFrame *Initialize a WaterLevelDataFrame*

Description

To initialize an object of class WaterLevelDataFrame this function should be used. It checks all the required input data and validates the final object.

Usage

```
WaterLevelDataFrame(
  river,
  time,
  gauging_stations = NULL,
  gauging_stations_missing = NULL,
```

```
  comment = NULL,
  id = NULL,
  station = NULL,
  station_int = NULL,
  w = NULL
)
```

Arguments

river	a required argument to fill the WaterLevelDataFrame-slot river. It has to be type character, has to have a length of one and can be **Elbe** or **Elbe_tidal** or **Rhine**.
time	a required argument to fill the WaterLevelDataFrame-slot time. It has to be type c("POSIXct","POSIXt"), has to have a length of one and must be in the temporal range between 1960-01-01 00:00:00 CET and now (Sys.time()) or be NA.
gauging_stations	a slot of class data.frame. gauging_stations has to be a data.frame with the following columns and column types: id (integer), gauging_station (character), uuid (character), km (numeric), km_qps (numeric), river (character), longitude (numeric), latitude (numeric), mw (numeric), pnp (numeric), w (numeric), wl (numeric), n_wls_below_w_do (integer), n_wls_above_w_do (integer), n_wls_below_w_up (integer), n_wls_above_w_up (integer), name_wl_below_w_do (character), name_wl_above_w_do (character), name_wl_below_w_up (character), name_wl_above_w_up (character), w_wl_below_w_do (numeric), w_wl_above_w_do (numeric), w_wl_below_w_up (numeric), w_wl_above_w_up (numeric), weight_up (numeric), weight_do (numeric).
gauging_stations_missing	an optional argument to fill the WaterLevelDataFrame-slot gauging_stations_missing. It has to be type character and usually contains a vector with names of gauging stations for which no water level information was available for the specified time. This argument is used by the functions waterLevel, waterLevelPegelonline, waterLevelFlys3 and waterLevelFlys3Seq.
comment	an optional argument to fill the WaterLevelDataFrame-slot comment. It has to be type character and is used by the functions WaterLevelDataFrame, waterLevel, waterLevelPegelonline, waterLevelFlys3 and waterLevelFlys3Seq.
id	an optional argument to hand over the row.names(wldf). id has to be type integer and has to have the same length as other optional arguments (station, station_int and w) forming the data.frame-component of a WaterLevelDataFrame.
station	an optional argument to hand over the stationing along the specified river. If specified, it has to be type numeric and has to have the same length as other optional arguments (id, station_int and w) forming the data.frame-component of a WaterLevelDataFrame. If both stationing arguments (station and station_int) are specified, all elements of station have to be equal to as.numeric(station_int / 1000). Minimum and maximum allowed values of station are river-specific: Elbe (km 0 - 585.7), Elbe_tidal (km 0 - 174.0), Rhine (km 336.2 - 865.7).

`station_int` an optional argument to hand over the stationing along the specified `river`. If specified, it has to be type `integer` and has to have the same length as other optional arguments (`id`, `station` and `w`) forming the `data.frame`-component of a WaterLevelDataFrame. If both stationing arguments (`station` and `station_int`) are specified, all elements of `station_int` have to be equal to `as.integer(station * 1000)`. Minimum and maximum allowed values of `station_int` are `river`-specific.

`w` an optional argument to hand over the water level information along the stationing of the specified `river` for a given `time`. If specified, it has to be type `numeric` and has to have the same length as other optional arguments (`id`, `station` and `station_int`) forming the `data.frame`-component of a WaterLevelDataFrame. If not specified, the respective WaterLevelDataFrame-column `w` can be computed by the functions `waterLevel`, `waterLevelPegelonline`, `waterLevelFlys3` and `waterLevelFlys3Seq`. Minimum and maximum allowed values of `w` are river-specific.

Value

The function produces an object of class WaterLevelDataFrame which might contain 1d water level data and information to recompute it.

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf <- waterLevel(wldf)
```

WaterLevelDataFrame-class

S4 class for 1d water level data

Description

The S4 class WaterLevelDataFrame is inherited from the S3 class `data.frame` and stores 1d water level information together with the official stationing along the German federal waterways Elbe and Rhine.

Details

In addition to the 1d water level data stored in the `data.frame` further slots contain necessary information used for or computed during the computation of water levels:

Slots

`.Data` contains the `data.frame` with at least three columns: `station`, `station_int` and `w`. The columns `station` and `station_int` represent the official stationing along the waterways in two different formats. They are totally exchangeable since `station <- as.numeric(station_int / 1000)` and `station_int <- as.integer(station * 1000)`. The column `w` represents the height of the water level relative to standard elevation zero (DHHN92). These first three columns are required, but further columns can be added.

`river` is a required slot clearly determining the location of a station. Possible values of `river` have to be type `character`, have to have a length of one.

`time` is a slot determining the time for which the water level has been computed. `time` has to be type `c("POSIXct","POSIXt")`, has to have a length of one and be in the range between `1960-01-01 00:00:00 CET` and now (`Sys.time()`) or `NA`.

`gauging_stations` possibly contains a `data.frame` with relevant information about gauging stations within the relevant `river` stretch and the closer surrounding up- and downstream of the relevant `river` stretch. It is usually filled by the functions `waterLevel` or `waterLevelPegelonline`.

`gauging_stations_missing` possibly contains a vector of type `character` with names of gauging stations for which no gauging data existed for the requested `time`. It is automatically filled by the functions `waterLevel`, `waterLevelPegelonline`, `waterLevelFlys3` and `waterLevelFlys3Seq`.

`comment` contains information on which function has been used to create (`WaterLevelDataFrame`) or compute (`waterLevel`, `waterLevelPegelonline`, `waterLevelFlys3` and `waterLevelFlys3Seq`) an object of class WaterLevelDataFrame.

`waterLevelFlood1` *Compute 1d water level data from the FLYS3 water level MQ and a gauging station according to the INFORM 3-method Flood1 (Flut1)*

Description

This function computes a 1d water level according to the INFORM flood duration method Flood1 (Flut1) and stores it as column `w` of an S4 object of type WaterLevelDataFrame. First the function obtains the reference water level MQ from `df.flys`. This reference water level is then shifted by the difference between measured water and the FLYS3 water level for MQ at the specified gauging station. Here it is provided mainly for historical reasons and more advanced functions like `waterLevel` or `waterLevelPegelonline` should be used.

Usage

```
waterLevelFlood1(wldf, gauging_station, w, uuid, shiny = FALSE)
```

Arguments

`wldf`	an object of class WaterLevelDataFrame.

gauging_station
: must be type character with a length of one. Permitted values are: 'SCHOENA', 'PIRNA', 'DRESDEN', 'MEISSEN', 'RIESA', 'MUEHLBERG', 'TORGAU', 'PRETZSCH-MAUKEN', 'ELSTER', 'WITTENBERG', 'COSWIG', 'VOCKERODE', 'ROSSLAU', 'DESSAU', 'AKEN', 'BARBY', 'SCHOENEBECK', 'MAGDEBURG-BUCKAU', 'MAGDEBURG-STROMBRUECKE', 'MAGDEBURG-ROTHENSEE', 'NIEGRIPP AP', 'ROGAETZ', 'TANGERMUENDE', 'STORKAU', 'SANDAU', 'SCHARLEUK', 'WITTENBERGE', 'MUEGGENDORF', 'SCHNACKENBURG', 'LENZEN', 'GORLEBEN', 'DOEMITZ', 'DAMNATZ', 'HITZACKER', 'NEU DARCHAU', 'BLECKEDE', 'BOIZENBURG', 'HOHNSTORF', 'ARTLENBURG', 'GEESTHACHT', 'IFFEZHEIM', 'PLITTERSDORF', 'MAXAU', 'PHILIPPSBURG', 'SPEYER', 'MANNHEIM', 'WORMS', 'NIERSTEIN-OPPENHEIM', 'MAINZ', 'OESTRICH', 'BINGEN', 'KAUB', 'SANKT GOAR', 'BOPPARD', 'BRAUBACH', 'KOBLENZ', 'ANDERNACH', 'OBERWINTER', 'BONN', 'KOELN', 'DUESSELDORF', 'RUHRORT', 'WESEL', 'REES', 'EMMERICH'.

w
: If the wldf does not supply a valid non-NA time slot, it is possible to execute the function with the help of this optional parameter. Otherwise getGaugingDataW or getPegelonlineW provide gauging data internally.

uuid
: must be type character with a length of one. Permitted values are: '7cb7461b-3530-4c01-8978-7f676b8f71ed', '85d686f1-55b2-4d36-8dba-3207b50901a7', '70272185-b2b3-4178-96b8-43bea330dcae', '24440872-5bd2-4fb3-8554-907b49816c49', 'b04b739d-7ffa-41ee-9eb9-95cb1b4ef508', '16b9b4e7-be14-41fd-941e-6755c97276cc', '83bbaedb-5d81-4bc6-9f66-3bd700c99c1f', 'f3dc8f07-c2bb-4b92-b0b0-4e01a395a2c6', 'c093b557-4954-4f05-8f5c-6c6d7916c62d', '070b1eb4-3872-4e07-b2e5-e25fd9251b93', '1ce53a59-33b9-40dc-9b17-3cd2a2414607', 'ae93f2a5-612e-4514-b5fd-9c8aecdd73c7', 'e97116a4-7d30-4671-8ba1-cdce0a153d1d', '1edc5fa4-88af-47f5-95a4-0e77a06fe8b1', '094b96e5-caeb-46d3-a8ee-d44182add069', '939f82ec-15a9-49c8-8828-dc2f8a2d49e2', '90bcb315-f080-41a8-a0ac-6122331bb4cf', 'b8567c1e-8610-4c2b-a240-65e8a74919fa', 'ccccb57f-a2f9-4183-ae88-5710d3afaefd', 'e30f2e83-b80b-4b96-8f39-fa60317afcc7', '3adf88fd-fd7a-41d0-84f5-1143c98a6564', '133f0f6c-2ca1-4798-9360-5b5f417dd839', '13e91b77-90f3-41a5-a320-641748e9c311', 'de4cc1db-51cb-4b62-bee2-9750cbe4f5c4', 'f4c55f77-ab80-4e00-bed3-aa6631aba074', 'e32b0a28-8cd5-4053-bc86-fff9c6469106', 'cbf3cd49-91bd-49cc-8926-ccc6c0e7eca4', '48f2661f-f9cb-4093-9d57-da2418ed656e', '550e3885-a9d1-4e55-bd25-34228bd6d988', 'c80a4f21-528c-4771-98d7-10cd591699a4', 'ac507f42-1593-49ea-865f-10b2523617c7', '6e3ea719-48b1-408a-bc55-0986c1e94cd5', 'c233674f-259a-4304-b81f-dce1f415d85b', 'a26e57c9-1cb8-4fca-ba80-9e02abc81df8', '67d6e882-b60c-40d3-975c-a6d7a2b4e40a', '6aa1cd8e-e528-4bcb-ba8e-705b6dcb7da2', '33e0bce0-13df-4ffc-be9d-f1a79e795e1c', 'd9289367-c8aa-4b6a-b1ad-857fec94c6bb', 'b3492c68-8373-4769-9b29-22f66635a478', '44f7e955-c97d-45c8-9ed7-19406806fb4c', 'b02be240-1364-4c97-8bb6-675d7d842332', '6b774802-fcb5-49ae-8ecb-ecaf1a278b1c', 'b6c6d5c8-e2d5-4469-8dd8-fa972ef7eaea', '88e972e1-88a0-4eb9-847c-0925e5999a46', '2cb8ae5b-c5c9-4fa8-bac0-bb724f2754f4', '57090802-c51a-4d09-8340-b4453cd0e1f5', '844a620f-f3b8-4b6b-8e3c-783ae2aa232a', 'd28e7ed1-3317-41c5-bec6-725369ed1171', 'a37a9aa3-45e9-4d90-9df6-109f3a28a5af', '665be0fe-5e38-43f6-8b04-02a93bdbeeb4', '0309cd61-90c9-470e-99d4-2ee4fb2c5f84', '1d26e504-7f9e-480a-b52c-5932be6549ab', '550eb7e9-172e-48e4-ae1e-d1b761b42223', '2ff6379d-d168-4022-8da0-16846d45ef9b', 'd6dc44d1-63ac-4871-b175-60ac4040069a', '4c7d796a-39f2-4f26-97a9-3aad01713e29', '5735892a-ec65-4b29-97c5-50939aa9584e', 'b45359df-c020-4314-adb1-d1921db642da',

'593647aa-9fea-43ec-a7d6-6476a76ae868', 'a6ee8177-107b-47dd-bcfd-30960ccc6e9c', '8f7e5f92-1153-4f93-acba-ca48670c8ca9', 'c0f51e35-d0e8-4318-afaf-c5fcbc29f4c1', 'f33c3cc9-dc4b-4b77-baa9-5a5f10704398', '2f025389-fac8-4557-94d3-7d0428878c86', '9598e4cb-0849-401e-bba0-689234b27644'.

`shiny` `logical` determing whether columns (`section`, `weight_x`, `weight_y`) relevant for the `plotShiny()`-function are appended to the resulting WaterLevelDataFrame.

Details

This function computes a water level based on the reference water level MQ from `df.flys`. Since the function only shifts this single reference water level to make it fit to the measured water level, no interpolation is needed. Therefore the `shiny` columns have constant values of `section <- 1`, `weight_x <- 1` and `weight_y <- shift`.

Value

An object of class WaterLevelDataFrame.

References

Rosenzweig S, Giebel H, Schleuter M (2011). "Ökologische Modellierungen für die Wasser- und Schifffahrtsverwaltung – Das integrierte Flussauenmodell INFORM in seiner neuesten Fassung (Version 3). Bundesanstalt für Gewässerkunde, Koblenz, Germany." doi:10.5675/bfg1667.

Bundesanstalt für Gewässerkunde (2016). "FLYS – Flusshydrologischer Webdienst." https://www.bafg.de/DE/5_Informiert/1_Portale_Dienste/FLYS/flys_node.html.

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf1 <- waterLevelFlood1(wldf, "ROSSLAU")
wldf2 <- waterLevelFlood1(wldf, "DESSAU")

wldf1$w - wldf2$w
```

`waterLevelFlood2` *Compute 1d water level data through linear interpolation with neighboring gauging stations according to the INFORM 3-method Flood2 (Flut2)*

Description

This function computes a 1d water level according to the INFORM flood duration method Flood2 (Flut2) and stores it as column `w` of an S4 object of type WaterLevelDataFrame. Flood2 is designed to enable water level computation between gauging stations along waterways without reference water levels, provided for example by FLYS3. The function uses neighboring gauging stations for linear interpolation of gauging station water levels along the selected river stretch. Here it is provided mainly for historical reasons and more advanced functions like `waterLevel` or `waterLevelPegelonline` should be used.

Usage

```
waterLevelFlood2(wldf, value = NULL, df = NULL)
```

Arguments

`wldf`	an object of class WaterLevelDataFrame.
`value`	an optional value of type `character`. Commonly available values are `c("MThw", "MTnw", "HThw", "NTnw", "HHW", "NNW", "MNW","MW", "MHW")` or a column supplied in `df`.
`df`	an optional object of type `data.frame`, which must contain the columns `gauging_station`, `river`, `longitude`, `latitude`, `km_csa`, `pnp` and finally a water level column named in `value`.

Details

This function computes a water level through simple linear interpolation of water levels at neighboring gauging stations. Historically it has been designed for rivers without 1d reference water levels provided by FLYS3 for `df.flys`. In the meantime it has been extended to linearly interpolate characteristic water levels queried from PEGELONLINE through `getPegelonlineCharacteristicValues` or supplied through the external data provided with `df`.

Value

An object of class WaterLevelDataFrame.

References

Rosenzweig S, Giebel H, Schleuter M (2011). "Ökologische Modellierungen für die Wasser- und Schifffahrtsverwaltung – Das integrierte Flussauenmodell INFORM in seiner neuesten Fassung (Version 3). Bundesanstalt für Gewässerkunde, Koblenz, Germany." doi:10.5675/bfg1667.

See Also

`getPegelonlineCharacteristicValues`

Examples

```
# using internal gauging data
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf1 <- waterLevelFlood2(wldf)

# using characteristic water levels queried from PEGELONLINE
wldf2 <- WaterLevelDataFrame(river   = "Elbe_tidal",
                            time    = as.POSIXct(NA),
                            station_int = as.integer(seq(0, 170000, 1000)))
wldf2 <- waterLevelFlood2(wldf2, "MThw")

# supply external water levels through a data.frame
df <- data.frame(gauging_station = c("test1", "test2"),
                 river = c("Elbe_tidal", "Elbe_tidal"),
                 longitude = c(4, 5),
                 latitude = c(4, 5),
                 km_csa = c(60, 40),
                 pnp = c(0, 0),
                 MThw = c(400, 450))
wldf3 <- WaterLevelDataFrame(river   = "Elbe_tidal",
                            time    = as.POSIXct(NA),
                            station_int = as.integer(seq(30000, 70000,
                                                         1000)))
wldf3 <- waterLevelFlood2(wldf3, "MThw", df)
```

`waterLevelFlys3` *Obtain 1d water level data from the FLYS3 database*

Description

Obtain 1d water level data from the FLYS3 database using either a predefined WaterLevelDataFrame or `river`, `from` and `to` arguments that enable the internal construction of a WaterLevelDataFrame. The internally constructed WaterLevelDataFrame contains stations every 0.1 km or 100 m between the given range of `from` and `to`.

Usage

```
waterLevelFlys3(wldf, name)

waterLevelFlys3Seq(river = c("Elbe", "Rhine"), name, from, to)
```

Arguments

`wldf`	an object of class WaterLevelDataFrame.

name	a string with the name of a stationary FLYS3 water level. It has to be type character, has to have a length of one and has to be an element of the river-specific names specified in Details.
river	a required argument to fill the WaterLevelDataFrame-slot river. It has to be type character, has to have a length of one and can be either **Elbe** or **Rhine**.
from	numeric or integer for the upstream station. It has to have a length of one and has to be within the river-specific possible station range specified in Details.
to	numeric or integer for the downstream station. It has to have the same type as from, a length of one and has to be within the river-specific possible station range specified in Details.

Details

Possible names of FLYS3 water levels and ranges of from and to are river-specific:

Elbe:

'0.5MNQ', 'MNQ', '0.5MQ', 'a', '0.75MQ', 'b', 'MQ', 'c', '2MQ', '3MQ', 'd', 'e', 'MHQ', 'HQ2', 'f', 'HQ5', 'g', 'h', 'HQ10', 'HQ15', 'HQ20', 'HQ25', 'HQ50', 'HQ75', 'HQ100', 'i', 'HQ150', 'HQ200', 'HQ300', 'HQ500'

Possible range of from and to: type numeric (km) 0 - 585.7, type integer (m) 0 - 585700.

Rhine:

'Ud=1', 'Ud=5', 'GlQ2012', 'Ud=50', 'Ud=80', 'Ud=100', 'Ud=120', 'Ud=183', 'MQ', 'Ud=240', 'Ud=270', 'Ud=310', 'Ud=340', 'Ud=356', 'Ud=360', 'MHQ', 'HQ2', 'HQ5', 'HQ5-10', 'HQ10', 'HQ10-20', '~HQ20', 'HQ20-50', 'HQ50', 'HQ50-100', 'HQ100', 'HQ100-200', 'HQ200', 'HQ200-ex', 'HQextr.'

Possible range of from and to: type numeric (km) 336.2 - 865.7, type integer (m) 336200 - 865700.

Both lists of water levels are ordered from low to high water levels.

Value

An object of class WaterLevelDataFrame.

References

Busch N, Hammer M (2009). "Einheitliche Grundlage für die Festlegung der Bemessungswasserspiegellagen der Elbe auf der frei fließenden Strecke in Deutschland." doi:10.5675/bfg1650.

HKV Hydrokontor (2014). "Erstellung eines SOBEK-River Modells für den Rhein von Iffezheim bis Pannerdense Kop als Weiterentwicklung bestehender SOBEK-RE Modelle."

Bundesanstalt für Gewässerkunde (2013). "FLYS goes WEB: Eröffnung eines neuen hydrologischen Fachdienstes in der BfG." doi:10.5675/BfG_Veranst_2013.4. https://doi.bafg.de/BfG/2013/Veranst4_2013.pdf.

Bundesanstalt für Gewässerkunde (2016). "FLYS – Flusshydrologischer Webdienst." https://www.bafg.de/DE/5_Informiert/1_Portale_Dienste/FLYS/flys_node.html.

See Also

df.flys, plotShiny

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf1 <- waterLevelFlys3(wldf, "MQ")

wldf2 <- waterLevelFlys3Seq("Elbe", "MQ", 257, 262)
```

---

waterLevelFlys3InterpolateX

Interpolate FLYS3 water levels for given stations

Description

Function to interpolate FLYS3 water levels for selected stations and return it with the structure of df.flys.

Usage

```
waterLevelFlys3InterpolateX(
  river = c("Elbe", "Rhine"),
  station = NULL,
  station_int = NULL
)
```

Arguments

river	a required argument to fill the WaterLevelDataFrame-slot river. It has to be type character, has to have a length of one and can be either **Elbe** or **Rhine**.
station	an optional argument to hand over the stationing along the specified river. If specified, it has to be type numeric and has to have the same length as other optional arguments (id, station_int and w) forming the data.frame-component of a WaterLevelDataFrame. If both stationing arguments (station and station_int) are specified, all elements of station have to be equal to as.numeric(station_int / 1000). Minimum and maximum allowed values of station are river-specific: Elbe (km 0 - 585.7), Rhine (km 336.2 - 865.7).
station_int	an optional argument to hand over the stationing along the specified river. If specified, it has to be type integer and has to have the same length as other optional arguments (id, station and w) forming the data.frame-component of a WaterLevelDataFrame. If both stationing arguments (station and station_int) are specified, all elements of station_int have to be equal to as.integer(station * 1000). Minimum and maximum allowed values of station_int are river-specific: Elbe (m 0 - 585700), Rhine (m 336200 - 865700).

Details

df.flys contains 1d water level data computed with SOBEK for every second hectometer (every 200 m). This function provides a way to interpolate the 30 stationary water levels for selected stations inbetween these hectometers and returns them with the data.frame-structure of the original dataset.

Value

An object of class data.frame with the structure of df.flys.

References

Busch N, Hammer M (2009). "Einheitliche Grundlage für die Festlegung der Bemessungswasserspiegellagen der Elbe auf der frei fließenden Strecke in Deutschland." doi:10.5675/bfg1650.

HKV Hydrokontor (2014). "Erstellung eines SOBEK-River Modells für den Rhein von Iffezheim bis Pannerdense Kop als Weiterentwicklung bestehender SOBEK-RE Modelle."

DELTARES (2018). "SOBEK." https://download.deltares.nl/en/sobek/.

See Also

df.flys

Examples

```
df.flys  <- waterLevelFlys3InterpolateX("Elbe", 257.1)
```

---

waterLevelFlys3InterpolateY

Compute a 1d water level dataset based on the FLYS3 algorythms

---

Description

Function to compute 1d water level information based on the original FLYS3 algorythms and store it as column w of an S4 object of type WaterLevelDataFrame.

Usage

```
waterLevelFlys3InterpolateY(wldf, gauging_station, w, uuid, shiny = FALSE)
```

Arguments

wldf	an object of class WaterLevelDataFrame.
gauging_station	must be type character with a length of one. Permitted values are: 'SCHOENA', 'PIRNA', 'DRESDEN', 'MEISSEN', 'RIESA', 'MUEHLBERG', 'TORGAU', 'PRETZSCH-MAUKEN', 'ELSTER', 'WITTENBERG', 'COSWIG', 'VOCKERODE', 'ROSSLAU', 'DESSAU', 'AKEN', 'BARBY', 'SCHOENEBECK', 'MAGDEBURG-BUCKAU', 'MAGDEBURG-STROMBRUECKE', 'MAGDEBURG-ROTHENSEE', 'NIEGRIPP AP', 'ROGAETZ', 'TANGERMUENDE', 'STORKAU', 'SANDAU', 'SCHARLEUK', 'WITTENBERGE', 'MUEGGENDORF', 'SCHNACKENBURG', 'LENZEN', 'GORLEBEN', 'DOEMITZ', 'DAMNATZ', 'HITZACKER', 'NEU DARCHAU', 'BLECKEDE', 'BOIZENBURG', 'HOHNSTORF', 'ARTLENBURG', 'GEESTHACHT', 'IFFEZHEIM', 'PLITTERSDORF', 'MAXAU', 'PHILIPPSBURG', 'SPEYER', 'MANNHEIM', 'WORMS', 'NIERSTEIN-OPPENHEIM', 'MAINZ', 'OESTRICH', 'BINGEN', 'KAUB', 'SANKT GOAR', 'BOPPARD', 'BRAUBACH', 'KOBLENZ', 'ANDERNACH', 'OBERWINTER', 'BONN', 'KOELN', 'DUESSELDORF', 'RUHRORT', 'WESEL', 'REES', 'EMMERICH'.
w	If the wldf does not supply a valid non-NA time slot, it is possible to execute the function with the help of this optional parameter. Otherwise getGaugingDataW or getPegelonlineW provide gauging data internally.
uuid	must be type character with a length of one. Permitted values are: '7cb7461b-3530-4c01-8978-7f676b8f71ed', '85d686f1-55b2-4d36-8dba-3207b50901a7', '70272185-b2b3-4178-96b8-43bea330dcae', '24440872-5bd2-4fb3-8554-907b49816c49', 'b04b739d-7ffa-41ee-9eb9-95cb1b4ef508', '16b9b4e7-be14-41fd-941e-6755c97276cc', '83bbaedb-5d81-4bc6-9f66-3bd700c99c1f', 'f3dc8f07-c2bb-4b92-b0b0-4e01a395a2c6', 'c093b557-4954-4f05-8f5c-6c6d7916c62d', '070b1eb4-3872-4e07-b2e5-e25fd9251b93', '1ce53a59-33b9-40dc-9b17-3cd2a2414607', 'ae93f2a5-612e-4514-b5fd-9c8aecdd73c7', 'e97116a4-7d30-4671-8ba1-cdce0a153d1d', '1edc5fa4-88af-47f5-95a4-0e77a06fe8b1', '094b96e5-caeb-46d3-a8ee-d44182add069', '939f82ec-15a9-49c8-8828-dc2f8a2d49e2', '90bcb315-f080-41a8-a0ac-6122331bb4cf', 'b8567c1e-8610-4c2b-a240-65e8a74919fa', 'ccccb57f-a2f9-4183-ae88-5710d3afaefd', 'e30f2e83-b80b-4b96-8f39-fa60317afcc7', '3adf88fd-fd7a-41d0-84f5-1143c98a6564', '133f0f6c-2ca1-4798-9360-5b5f417dd839', '13e91b77-90f3-41a5-a320-641748e9c311', 'de4cc1db-51cb-4b62-bee2-9750cbe4f5c4', 'f4c55f77-ab80-4e00-bed3-aa6631aba074', 'e32b0a28-8cd5-4053-bc86-fff9c6469106', 'cbf3cd49-91bd-49cc-8926-ccc6c0e7eca4', '48f2661f-f9cb-4093-9d57-da2418ed656e', '550e3885-a9d1-4e55-bd25-34228bd6d988', 'c80a4f21-528c-4771-98d7-10cd591699a4', 'ac507f42-1593-49ea-865f-10b2523617c7', '6e3ea719-48b1-408a-bc55-0986c1e94cd5', 'c233674f-259a-4304-b81f-dce1f415d85b', 'a26e57c9-1cb8-4fca-ba80-9e02abc81df8', '67d6e882-b60c-40d3-975c-a6d7a2b4e40a', '6aa1cd8e-e528-4bcb-ba8e-705b6dcb7da2', '33e0bce0-13df-4ffc-be9d-f1a79e795e1c', 'd9289367-c8aa-4b6a-b1ad-857fec94c6bb', 'b3492c68-8373-4769-9b29-22f66635a478', '44f7e955-c97d-45c8-9ed7-19406806fb4c', 'b02be240-1364-4c97-8bb6-675d7d842332', '6b774802-fcb5-49ae-8ecb-ecaf1a278b1c', 'b6c6d5c8-e2d5-4469-8dd8-fa972ef7eaea', '88e972e1-88a0-4eb9-847c-0925e5999a46', '2cb8ae5b-c5c9-4fa8-bac0-bb724f2754f4', '57090802-c51a-4d09-8340-b4453cd0e1f5', '844a620f-f3b8-4b6b-8e3c-783ae2aa232a', 'd28e7ed1-3317-41c5-bec6-725369ed1171', 'a37a9aa3-45e9-4d90-9df6-109f3a28a5af', '665be0fe-5e38-43f6-8b04-02a93bdbeeb4', '0309cd61-90c9-470e-99d4-2ee4fb2c5f84', '1d26e504-7f9e-480a-b52c-5932be6549ab',

'550eb7e9-172e-48e4-ae1e-d1b761b42223', '2ff6379d-d168-4022-8da0-16846d45ef9b', 'd6dc44d1-63ac-4871-b175-60ac4040069a', '4c7d796a-39f2-4f26-97a9-3aad01713e29', '5735892a-ec65-4b29-97c5-50939aa9584e', 'b45359df-c020-4314-adb1-d1921db642da', '593647aa-9fea-43ec-a7d6-6476a76ae868', 'a6ee8177-107b-47dd-bcfd-30960ccc6e9c', '8f7e5f92-1153-4f93-acba-ca48670c8ca9', 'c0f51e35-d0e8-4318-afaf-c5fcbc29f4c1', 'f33c3cc9-dc4b-4b77-baa9-5a5f10704398', '2f025389-fac8-4557-94d3-7d0428878c86', '9598e4cb-0849-401e-bba0-689234b27644'.

`shiny` `logical` determing whether columns (`section`, `weight_x`, `weight_y`) relevant for the `plotShiny()`-function are appended to the resulting WaterLevelDataFrame.

Value

An object of class WaterLevelDataFrame.

References

Busch N, Hammer M (2009). "Einheitliche Grundlage für die Festlegung der Bemessungswasserspiegellagen der Elbe auf der frei fließenden Strecke in Deutschland." doi:10.5675/bfg1650.

HKV Hydrokontor (2014). "Erstellung eines SOBEK-River Modells für den Rhein von Iffezheim bis Pannerdense Kop als Weiterentwicklung bestehender SOBEK-RE Modelle."

DELTARES (2018). "SOBEK." https://download.deltares.nl/en/sobek/.

See Also

`df.flys`

Examples

```
# waterLevelFlys3InterpolateY
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 263, 0.1))
wldf <- waterLevelFlys3InterpolateY(wldf, "ROSSLAU", w = 137)
```

`[.WaterLevelDataFrame` *Extract or replace parts of a WaterLevelDataFrame*

Description

Extract or replace subsets of the `.Data` slot of an object of class WaterLevelDataFrame.

Usage

```
## S4 method for signature 'WaterLevelDataFrame'
x[i, j]

## S4 replacement method for signature 'WaterLevelDataFrame,ANY,ANY,data.frame'
x[i, j] <- value
```

Arguments

x	object of class WaterLevelDataFrame.
i, j	elements to extract or replace. For [, these are numeric or character or empty. Numeric values are coerced to integer as if by as.integer. For replacement by [, a logical matrix is allowed.
value	A suitable replacement value: it will be repeated a whole number of times if necessary and it may be coerced: see the Coercion section. If NULL, deletes the column if a single column is selected.

Details

For details see [.data.frame.

Value

A new object of class WaterLevelDataFrame is returned. Since the extraction or replacement acts only on the .Data-slot of the object, all other slots remain unchanged.

See Also

[.data.frame

Examples

```
wldf <- WaterLevelDataFrame(river   = "Elbe",
                            time    = as.POSIXct("2016-12-21"),
                            station = seq(257, 262, 0.1))
wldf <- wldf[which(wldf$station >= 259 & wldf$station <= 261), ]
```

Index