Package ‘WaterML’
August 29, 2016

Type Package
Title Fetch and Analyze Data from 'WaterML' and 'WaterOneFlow' Web Services
Version 1.7.1
Description Lets you connect to any of the Consortium of Universities for the Advancement of Hydrologic Sciences, Inc. (CUAHSI) Water Data Center 'WaterOneFlow' web services and read any 'WaterML' hydrological time series data file. To see list of available web services, see <http://hiscentral.cuahsi.org>. All versions of 'WaterML' (1.0, 1.1 and 2.0) and both types of the web service protocol ('SOAP' and 'REST') are supported. The package has six data download functions: GetServices(): show all public web services from the HIS Central Catalog. HISCentral_GetSites() and HISCentral_GetSeriesCatalog(): search for sites or time series from the HIS Central catalog based on geographic bounding box, server, or keyword. GetVariables(): Show a data.frame with all variables on the server. GetSites(): Show a data.frame with all sites on the server. GetSiteInfo(): Show what variables, methods and quality control levels are available at the specific site. GetValues(): Given a site code, variable code, start time and end time, fetch a data.frame of all the observation time series data values. The GetValues() function can also parse 'WaterML' data from a custom URL or from a local file. The package also has five data upload functions: AddSites(), AddVariables(), AddMethods(), AddSources(), and AddValues(). These functions can be used for uploading data to a 'HydroServer Lite' Observations Data Model (ODM) database via the 'JSON' data upload web service interface.

License MIT + file LICENSE
Depends R (>= 3.0)
Imports stats, XML, RJSONIO, httr
LazyData true
URL https://github.com/jirikadlec2/waterml
Suggests knitr
VignetteBuilder knitr
RoxygenNote 5.0.1
NeedsCompilation no
Author Jiri Kadlec [aut, cre]
Maintainer  Jiri Kadlec <jirikadlec2@gmail.com>
Repository  CRAN
Date/Publication  2016-03-17 23:19:55

R topics documented:

WaterML-package .................................................. 2
AddMethods .......................................................... 3
AddSites .............................................................. 4
AddSources ........................................................... 5
AddValues ............................................................ 6
AddVariables ......................................................... 7
GetServices .......................................................... 8
GetSiteInfo ........................................................... 9
GetSites .............................................................. 11
GetValues ............................................................ 12
GetVariables ......................................................... 13
HISCentral_GetSeriesCatalog ..................................... 14
HISCentral_GetSites ................................................ 16
MakeSOAPEnvelope .................................................. 17
WaterMLVersion ...................................................... 18
WaterOneFlowNamespace .......................................... 19
WaterOneFlowVersion .............................................. 19

Index 21

WaterML-package  Fetch and analyze data from WaterML or CUAHSI WaterOneFlow web service

Description

Package: WaterML
Type: Package
License: MIT
LazyLoad: yes

Details

Fetch and analyze data from WaterML or CUAHSI WaterOneFlow web service.

Please see https://github.com/jirikadlec2/waterml for more information.
Description

This function adds a table of methods to HydroServer Lite. The input must be a data.frame with all required ODM 'method' fields. NOTE: this only works with HydroServer Lite that implements the JSON API. You must specify a valid server URL, user name, and password for the HydroServer. The examples here use the 'sandbox' HydroServer on http://worldwater.byu.edu/app/ with the username: admin and password: password.

Usage

AddMethods(server, username, password, methods)

Arguments

server The URL of the web service ending with /services or with ?wsdl, for example: http://worldwater.byu.edu/app/index.php/default/services/cuahsi_1_1.asmx?wsdl alternatively you can specify the JSON API url like: http://worldwater.byu.edu/app/index.php/default/services/api/
username The valid HydroServer Lite username, for example "admin"
password The valid HydroServer Lite password, for example "password"
methods The valid table of methods. This table must have the following columns: MethodDescription, MethodLink.

Value

A table of the added methods, with two extra columns: MethodID (the ID assigned by the server), Status (the status showing if the method was added: OK or Error). If the status is Error, then the Error message with reason why the method could not be added is also shown.

Examples

user <- "admin"
pass <- "password"
server <- "http://worldwater.byu.edu/app/index.php/default/services/cuahsi_1_1.asmx"

# make random site codes
random_description <- sprintf("R Test Method %04d", sample(1:10000, 1))
random_link <- "http://example.com"
my_methods <- data.frame(
    MethodDescription = random_description,
    MethodLink = random_link
)

added_methods <- AddMethods(server, username=user, password=pass, 
    methods=my_methods)
AddSites

Description

This function adds a table of sites to HydroServer Lite. The input must be a data.frame with all required ODM site fields. NOTE: this only works with HydroServer Lite that implements the JSON API. You must specify a valid server url, user name, and password for the HydroServer. The examples here use the 'sandbox' HydroServer on http://worldwater.byu.edu/app/ with the username: admin and password: password.

Usage

AddSites(server, username, password, sites)

Arguments

server
The URL of the web service ending with /services or with ?wsdl, for example: http://worldwater.byu.edu/app/index.php/default/services/cuahsi_1_1.asmx?wsdl alternatively you can specify the JSON API url like: http://worldwater.byu.edu/app/index.php/default/services/api/

username
The valid HydroServer Lite username, for example "admin"

password
The valid HydroServer Lite password, for example "password"

sites
The valid table of sites. This table must have the following 4 columns: Site-Code, SiteName, Latitude, Longitude. It can also have the optional columns: Elevation, SiteType, State, County, Comments.

Value

A table of the added sites, with two extra columns: SiteID (the ID assigned by the server), Status (the status showing if the site was added: OK or Error). If the status is Error, then the Error message with reason why the site could not be added is also shown.

Examples

user <- "admin"
pass <- "password"
server <- "http://worldwater.byu.edu/app/index.php/default/services/cuahsi_1_1.asmx"
#make random site codes
random_codes = sprintf("%04d", sample(1:10000, 2))
random_names = paste("R"," Upload", random_codes)
random_lats = runif(2, 35.0, 49.0) #two random latitudes inside U.S
random_lons = runif(2, -110.0, -70.0) #random longitudes inside U.S
my_sites <- data.frame(SiteCode=random_codes, SiteName=random_names,
                        Latitude=random_lats, Longitude=random_lons)

added_sites <- AddSites(server, username=user, password=pass, sites=my_sites)
AddSources

Description

This function adds a table of sources to HydroServer Lite. The input must be a data.frame with all required ODM 'Source' fields. NOTE: this only works with HydroServer Lite that implements the JSON API. You must specify a valid server url, user name, and password for the HydroServer. The examples here use the 'sandbox' HydroServer on http://worldwater.byu.edu/app/ with the username: admin and password: password.

Usage

addsources(server, username, password, sources)

Arguments

server

The URL of the web service ending with /services or with ?wsdl, for example:
http://worldwater.byu.edu/app/index.php/default/services/cuahsi_1_1.asmx?wsdl
alternatively you can specify the JSON API url like: http://worldwater.byu.edu/app/index.php/default/services/api/

username

The valid HydroServer Lite username, for example "admin"

password

The valid HydroServer Lite password, for example "password"

sources

The valid table of sources. This table must have the following columns: Organization, Description, SourceLink, ContactName, ContactPhone, ContactEmail, Address, City, State, Zipcode, Citation, MetadataID.

Value

A table of the added sources, with two extra columns: SourceID (the ID assigned by the server), Status (the status showing if the source was added: OK or Error). If the status is Error, then the Error message with reason why the source could not be added is also shown.

Examples

user <- "admin"
password <- "password"
server <- "http://worldwater.byu.edu/app/index.php/default/services/cuahsi_1_1.asmx"
make random source codes
random_id = sample(1:10000, size = 1)
random_name = paste("R test source", random_id)
my_sources <- data.frame(
  Organization = random_name,
  Description = paste("Uploaded from R:", random_name),
  SourceLink = paste("http://", random_id, sep = ""),
  ContactName = random_name,
  ContactPhone = "012-345-6789",
  ContactEmail = "test<at>gmail.com",
  Address = random_name,
City = random_name,
State = random_name,
Zipcode = random_id * 10,
Citation = paste("Uploaded from R as a test:", random_name),
MetadataID = 10
)

added_sources <- AddSources(server, username=user, password=pass,
                            sources=my_sources)

Description

This function adds a table of data values to HydroServer Lite. The input must be a data.frame
with Time and DataValue fields. The Time field must be POSIXct format and DataValue must be
numeric format. It is also required to enter a valid SiteID, VariableID, SourceID, MethodID and
QualityControlLevelID. New data values shall be inserted only if the SiteID, VariableID, SourceID,
MethodID and QualityControlLevelID entries already exist in the HydroServer. NOTE: this only
works with HydroServer Lite that implements the JSON API. You must specify a valid server url,
user name, and password for the HydroServer. The examples here use the 'sandbox' HydroServer
on http://worldwater.byu.edu/app/ with the username: admin and password: password.

Usage

AddValues(server, username, password, site, variable, methodID, sourceID,
           qualityControl, values)

Arguments

text

server
username
password
site
variable
methodID
sourceID
qualityControl
values

The URL of the web service ending with /services or with ?wsdl, for example:
http://worldwater.byu.edu/app/index.php/default/services/cuahsi_1_1.asmx?wsdl
Alternatively you can specify the JSON API url like: http://worldwater.byu.edu/app/index.php/default/service
The valid HydroServer Lite username, for example "admin"
The valid HydroServer Lite password, for example "password"
The valid SiteID
The valid VariableID
The valid MethodID
The valid SourceID
The valid QualityControlLevelID
The valid table of data values. This table must have the following columns:
Time (POSIXct), DataValue (numeric).
AddVariables

Value

Status (the status showing if the values were added: OK or Error). If the status is Error, then the Error message with reason why the values could not be added is also shown.

Examples

```r
user <- "admin"
pass <- "password"
server <- "http://worldwater.byu.edu/app/index.php/default/services/cuahsi_1_1.asmx"
sourceID = 15
qualityID = 1
variableID = 43
siteID = 170
methodID = 10
random_times <- sort(Sys.time() + runif(3, 0, 10)*60)
random_values <- runif(3, 0, 100)
my_values <- data.frame(Time=random_times, DataValue=random_values)

status <- AddValues(server, username=user, password=pass,
                     site=siteID, variable=variableID,
                     methodID=methodID, sourceID=sourceID,
                     qualityControl=qualityID, values=my_values)
```

---

AddVariables

Description

This function adds a table of variables to HydroServer Lite. The input must be a data.frame with all required ODM 'variable' fields NOTE: this only works with HydroServer Lite that implements the JSON API. you must specify a valid server url, user name, and password for the HydroServer. The examples here use the 'sandbox' HydroServer on http://worldwater.byu.edu/app/ with the username: admin and password: password.

Usage

```r
AddVariables(server, username, password, variables)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server</td>
<td>The URL of the web service ending with /services or with ?wsdl, for example: <a href="http://worldwater.byu.edu/app/index.php/default/services/cuahsi_1_1.asmx?wsdl">http://worldwater.byu.edu/app/index.php/default/services/cuahsi_1_1.asmx?wsdl</a> alternatively you can specify the JSON API url like: <a href="http://worldwater.byu.edu/app/index.php/default/services/api/">http://worldwater.byu.edu/app/index.php/default/services/api/</a></td>
</tr>
<tr>
<td>username</td>
<td>The valid HydroServer Lite username, for example &quot;admin&quot;</td>
</tr>
<tr>
<td>password</td>
<td>The valid HydroServer Lite password, for example &quot;password&quot;</td>
</tr>
</tbody>
</table>
variables The valid table of variables. This table must have the following columns: VariableCode, VariableName, Speciation, VariableUnitsID, SampleMedium, ValueType, IsRegular, TimeSupport, TimeUnitsID, DataType, GeneralCategory, NoDataValue. NOTE that the values of these fields must be in the CUAHSI controlled vocabulary.

Value

A table of the added variables, with two extra columns: VariableID (the ID assigned by the server), Status (the status showing if the variable was added: OK or Error). If the status is Error, then the Error message with reason why the variable could not be added is also shown.

Examples

user <- "admin"
pass <- "password"
server <- "http://worldwater.byu.edu/app/index.php/default/services/cuahsi_1_1.asmx"
# make random site codes
random_code = sprintf("R-%04d",sample(1:10000, 1))
my_variables <- data.frame(
  VariableCode = random_code,
  VariableName = "Color",
  Speciation = "Not Applicable",
  VariableUnitsID = 109,
  SampleMedium = "Groundwater",
  ValueType = "Sample",
  IsRegular = 1,
  TimeSupport = 0,
  TimeUnitsID = 100,
  DataType = "Average",
  GeneralCategory = "Hydrology",
  NoDataValue = -9999
)

added_variables <- AddVariables(server, username=user, password=pass,
  variables=my_variables)

GetServices

Description

This function gets the table of web services from the HIS Central catalog

Usage

GetServices()

Examples

GetServices()
GetSiteInfo

Description
This function gets the table variables measured at a specific site from the WaterML web service

Usage
GetSiteInfo(server, siteCode)

Arguments

server The URL of the web service ending with .asmx or .wsdl, for example: http://worldwater.byu.edu/app/index.php/rushvalley/services/cuahsi_1_1.asmx?WSDL
siteCode The full site code, for example: default:Ru5BMMA. To get a list of available site codes, see GetSites() function and use the FullSiteCode field.

Value

a data.frame of data values with the following columns:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>character</td>
<td>The site ID in the original database</td>
</tr>
<tr>
<td>SiteID</td>
<td>character</td>
<td>The name of the site</td>
</tr>
<tr>
<td>SiteName</td>
<td>character</td>
<td>A short unique code of the site</td>
</tr>
<tr>
<td>SiteCode</td>
<td>character</td>
<td>The complete unique code of the site</td>
</tr>
<tr>
<td>FullSiteCode</td>
<td>numeric</td>
<td>The WGS84 latitude in decimal degrees</td>
</tr>
<tr>
<td>in the format</td>
<td>numeric</td>
<td>The WGS84 longitude in decimal degrees</td>
</tr>
<tr>
<td>NETWORK:CODE, for</td>
<td>numeric</td>
<td>The elevation of the site above sea level in meters</td>
</tr>
<tr>
<td>example SNOTEL:879.</td>
<td>character</td>
<td>Only for sites in the USA: the state of the site</td>
</tr>
<tr>
<td>Latitude</td>
<td>character</td>
<td>Only for sites in the USA: The county of the site</td>
</tr>
<tr>
<td>Longitude</td>
<td>character</td>
<td>Additional comments about the sites</td>
</tr>
<tr>
<td>Elevation</td>
<td>character</td>
<td>Short code of the variable</td>
</tr>
<tr>
<td>State</td>
<td>character</td>
<td>The full variable code, for example: SNOTEL:SIW</td>
</tr>
<tr>
<td>County</td>
<td>character</td>
<td>The name of the variable</td>
</tr>
<tr>
<td>Comments</td>
<td>character</td>
<td>the type of observation:</td>
</tr>
<tr>
<td>(note: this field</td>
<td>character</td>
<td>the aggregate data type:</td>
</tr>
<tr>
<td>is often empty)</td>
<td></td>
<td>the general category of the measurements:</td>
</tr>
<tr>
<td>VariableCode</td>
<td>character</td>
<td>the sample medium:</td>
</tr>
<tr>
<td>FullVariableCode</td>
<td>character</td>
<td>The name of the measurement units</td>
</tr>
</tbody>
</table>
GetSiteInfo

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UnitType</td>
<td>character</td>
<td>the type of the measurement units</td>
</tr>
<tr>
<td>UnitAbbreviation</td>
<td>character</td>
<td>The abbreviation of the measurement units</td>
</tr>
<tr>
<td>NoDataValue</td>
<td>numeric</td>
<td>The value that indicates missing data</td>
</tr>
<tr>
<td>IsRegular</td>
<td>boolean</td>
<td>TRUE if the measurements are regular, FALSE otherwise</td>
</tr>
<tr>
<td>TimeUnitName</td>
<td>character</td>
<td>The name of the time units</td>
</tr>
<tr>
<td>TimeUnitAbbreviation</td>
<td>character</td>
<td>The time units abbreviation</td>
</tr>
<tr>
<td>TimeSupport</td>
<td>character</td>
<td>The length of the time period over which one measurement is taken</td>
</tr>
<tr>
<td>Speciation</td>
<td>character</td>
<td>The chemical sample speciation</td>
</tr>
<tr>
<td>methodID</td>
<td>character</td>
<td>The ID of the sensor or measurement method</td>
</tr>
<tr>
<td>methodCode</td>
<td>character</td>
<td>The code of the sensor or measurement method. Usually the same as methodID.</td>
</tr>
<tr>
<td>methodDescription</td>
<td>character</td>
<td>The description of the sensor or measurement method.</td>
</tr>
<tr>
<td>methodLink</td>
<td>character</td>
<td>The hyperlink of the website</td>
</tr>
<tr>
<td>sourceID</td>
<td>character</td>
<td>The ID of the data source or author</td>
</tr>
<tr>
<td>organization</td>
<td>character</td>
<td>The name of the organization collecting the data</td>
</tr>
<tr>
<td>sourceDescription</td>
<td>character</td>
<td>The description of organization collecting the data</td>
</tr>
<tr>
<td>citation</td>
<td>character</td>
<td>Instruction how to cite the data</td>
</tr>
<tr>
<td>qualityControlLevelID</td>
<td>character</td>
<td>The ID of the quality control level. Usually 0 means raw data and 1 means</td>
</tr>
<tr>
<td>qualityControlLevelCode</td>
<td>character</td>
<td>quality controlled data</td>
</tr>
<tr>
<td>qualityControlLevelDefinition</td>
<td>character</td>
<td>The quality control level definition.</td>
</tr>
<tr>
<td>valueCount</td>
<td>character</td>
<td>The number of observations in this time series</td>
</tr>
<tr>
<td>beginDateTime</td>
<td>POSIXct</td>
<td>The local date and time of the first available observation in this time</td>
</tr>
<tr>
<td>endDateTime</td>
<td>POSIXct</td>
<td>series.</td>
</tr>
<tr>
<td>beginDateTimeUTC</td>
<td>POSIXct</td>
<td>The UTC date and time of the first available observation in this time series</td>
</tr>
<tr>
<td>endDateTimeUTC</td>
<td>POSIXct</td>
<td>The UTC date and time of the last available observation in this time series</td>
</tr>
</tbody>
</table>

The output data.frame also has attributes with information about the status: download.time, parse.time, download.status, parse.status These attributes can be used for troubleshooting WaterOneFlow/WaterML server errors. If parse status is "NO_SERIES_FOUND", then this site doesn’t have any available data.

Examples

```r
server <- "http://worldwater.byu.edu/app/index.php/rushvalley/services/cuahsi_1_1.asmx"
siteInfo <- GetSiteInfo(server, siteCode="default:RuSBMMA")
```
**GetSites**

**Description**

This function gets the table of sites from the WaterML web service.

**Usage**

```r
GetSites(server, west = NULL, south = NULL, east = NULL, north = NULL)
```

**Arguments**

- `server`: The URL of the web service ending with .WSDL, for example: http://icewater.usu.edu/MudLake/cuahsi_1_0.asmx?WSDL; alternatively this can be the REST URL to get the sites.
- `west`: Optional parameter: The west longitude of the geographic bounding box in decimal degrees. Allowed values are between -180.0 and +180.0.
- `south`: Optional parameter: The south latitude of the geographic bounding box in decimal degrees. Allowed values are between -90.0 and +90.0.
- `east`: Optional parameter: The east longitude of the geographic bounding box in decimal degrees. Allowed values are between -180.0 and +180.0.
- `north`: Optional parameter: The north latitude of the geographic bounding box in decimal degrees. Allowed values are between -90.0 and +90.0.

**Value**

A data.frame of sites. The data.frame has the following columns:

- **SiteID**: The site ID in the original database
- **SiteName**: The name of the site
- **SiteCode**: A short unique code of the site
- **FullSiteCode**: The complete unique code of the site in the format NETWORK:CODE. Use this value in the GetSiteInfo and GetValues functions
- **Latitude**: The WGS84 latitude in decimal degrees
- **Longitude**: The WGS84 longitude in decimal degrees
- **Elevation**: The elevation of the site above sea level in meters
- **State**: Only for sites in the USA: the state of the site
- **County**: Only for sites in the USA: The county of the site
- **Comments**: Additional comments about the sites (note: this field is often empty)

The output data.frame also has attributes with information about the status: download.time, parse.time, download.status, parse.status. These attributes can be used for troubleshooting WaterOneFlow/WaterML server errors.
Examples

#Getting all sites from a service
sites <- GetSites("http://icewater.usu.edu/MudLake/cuahsi_1_0.asmx?WSDL")

#Getting a subset of sites restricted by geographical area
server <- "http://drought.usu.edu/usbrrreservoirs/cuahsi_1_1.asmx?WSDL"
sites_subset <- GetSites(server, west=-113.0, south=35.0, east=110.0, north=40.0)

---

**GetValues**

### Description

This function gets the time series data values from the WaterML web service

### Usage

```r
GetValues(server, siteCode = NULL, variableCode = NULL, startDate = NULL,
           endDate = NULL, methodID = NULL, sourceID = NULL, qcID = NULL,
           daily = NULL)
```

### Arguments

- `server`: The URL of the web service, for example: http://worldwater.byu.edu/interactive/rushvalley/services/index.php/cuahsi_1_1.asmx?WSDL. This can be also a custom REST URL or the file name of the WaterML file.

- `siteCode`: The site code. To get a list of available site codes, see GetSites() function and use the FullSiteCode field.

- `variableCode`: The variable code. To get a list of possible variable codes, see GetVariables() function and use the FullVariableCode field.

- `startDate`: (optional) The start date in "yyyy-mm-dd" format.

- `endDate`: (optional) The end date in "yyyy-mm-dd" format.

- `methodID`: (optional) The ID of the observation method. To get a list of possible method IDs, see methodID column in the output of GetSiteInfo(). If methodID is not specified, then the observations in the output data.frame won’t be filtered by method.

- `sourceID`: (optional) The ID of the source. To get a list of possible source IDs, see sourceID column in the output of GetSiteInfo(). If sourceID is not specified, then the observations in the output data.frame won’t be filtered by source.

- `qcID`: (optional) The ID of the quality control level. Typically 0 is used for raw data and 1 is used for quality controlled data. To get a list of possible quality control level IDs, see QualityControlLevelID column in the output of GetSiteInfo(). If qcID is not specified, then the observations in the output data.frame won’t be filtered by quality control level.

- `daily`: (optional) If you set daily="max", daily="min" or daily="mean", then the data values are aggregated to daily time step.
Value

A data.frame of data values with the following columns:

- **time**: The local date/time of the observation. The data type is POSIXct. POSIXct is a data type in R for storing time.
- **DataValue**: The observed data value
- **UTCOffset**: The difference between local time and UTC time in hours
- **CensorCode**: The code for censored observations. Possible values are nc (not censored), gt (greater than), lt (less than), nd (non-detect), pnq (present but not quantified)
- **DateTimeUTC**: The UTC time of the observation. The data type is POSIXct. POSIXct is a special data type in R for storing time.
- **MethodCode**: The code of the method or instrument used for the observation
- **SourceCode**: The code of the data source
- **QualityControlLevelCode**: The code of the quality control level. Possible values are -9999 (Unknown), 0 (Raw data), 1 (Quality controlled data), 2 (Derived products), 3 (Interpreted products), 4 (Knowledge products)

The output data.frame also has attributes with information about the status: download.time, parse.time, download.status, parse.status. These attributes can be used for troubleshooting WaterOneFlow/WaterML server errors. If parse status is "NO_VALUES_FOUND", then this time series doesn’t have any available data for the selected time period.

Examples

```r
# example 1: Get Values from a known site and variable from RushValley server
v1 <- GetValues("http://worldwater.byu.edu/app/index.php/rushvalley/services/cuahsi_1_1.asmx?WSDL", site="RuSBMMA", variable="SR5_Nr_NDVI", startDate="2014-11-01", endDate="2014-11-02", daily="max")

# example 2: Get values from an external REST URL (in this case the Provo USGS NWIS site id 10163000)
url <- "http://waterservices.usgs.gov/nwis/dv/?format=waterml,1.1&sites=10163000&parameterCd=00060"
v2 <- GetValues(url)

# example 3: Get values from WaterML 2.0 file and show year, month, day
url2 <- "http://www.waterml2.org/KiWIS-WML2-Example.wml"
waterml2_data <- GetValues(url2)
waterml2_data$year <- strftime(waterml2_data$time, "%Y")
waterml2_data$month <- strftime(waterml2_data$time, "%M")
waterml2_data$day <- strftime(waterml2_data$time, "%d")
```

Description

This function gets the table of variables from the WaterML web service.
HISCentral_GetSeriesCatalog

Usage

    GetVariables(server)

Arguments

    server  The URL of the web service ending with ?WSDL, for example: http://worldwater.byu.edu/app/index.php/rushvalley/services/cuahsi_1_1.asmx?WSDL

Value

    a data.frame of variables with the following columns:

    VariableCode character Short code of the variable
    FullVariableCode character The full variable code, for example: SNOTEL:897. Use this value as the variableCode parameter in GetValues() function.
    VariableName character The name of the variable
    ValueType character the type of observation: Field Observation or Derived Value
    DataType character the aggregate data type: Average, Continuous, Sporadic..
    GeneralCategory character the general category of the measurements: Climate, Water Quality..
    SampleMedium character the sample medium, for example water, atmosphere, soil..
    UnitName character The name of the measurement units
    UnitType character the type of the measurement units
    UnitAbbreviation character The abbreviation of the measurement units (m, cm, in..)
    NoDataValue numeric The value that indicates missing data
    IsRegular boolean TRUE if the measurements are regular, FALSE otherwise
    TimeUnitName character The name of the time units
    TimeUnitAbbreviation character The time units abbreviation
    TimeSupport character The length of the time period over which one measurement is taken
    Speciation character The chemical sample speciation (as nitrogen, as phosphorus..)

Examples

    GetVariables("http://worldwater.byu.edu/app/index.php/rushvalley/services/cuahsi_1_1.asmx?WSDL")
Arguments

- **west**: The west longitude of the geographic bounding box in decimal degrees. Allowed values are between -180.0 and +180.0
- **south**: The south latitude of the geographic bounding box in decimal degrees. Allowed values are between -90.0 and +90.0
- **east**: The east longitude of the geographic bounding box in decimal degrees. Allowed values are between -180.0 and +180.0
- **north**: The north latitude of the geographic bounding box in decimal degrees. Allowed values are between -90.0 and +90.0
- **serviceID** (optional): The ID of the service on HIS Central. To get the service ID, use the id column in the output of the GetServices() function.
- **keyword** (optional): The concept keyword (common name of variable) for searching the sites on HIS Central. Examples include Temperature, Precipitation, Snow Depth,... If the Keyword is not specified then sites with any variable will be returned.
- **beginDate** (optional): The begin date of the observations in yyyy-mm-dd format.
- **endDate** (optional): The end date of the observations in yyyy-mm-dd format.

Value

A data.frame of series catalog entries. The data.frame has the following columns:

- **ServiceCode**: The code of the HydroServer
- **ServiceURL**: The URL of the server. Use this as the server parameter in GetValues() function.
- **FullSiteCode**: The complete unique code of the site in the format NETWORK:CODE. Use this value as the siteCode parameter in the GetValues function.
- **FullVariableCode**: The complete unique code of the site in the format VOCABULARY:CODE. Use this value as the variableCode parameter in the GetValues function.
- **BeginDateTime**: The local date/time of the first observation of the time series in POSIXct format.
- **EndDateTime**: The local date/time of the last observation of the time series in POSIXct format.
- **ValueCount**: The number of measurements in the time series
- **SiteName**: The name of the site.
- **Latitude**: The WGS84 latitude of the site in decimal degrees
- **Longitude**: The WGS84 longitude of the site in decimal degrees
- **DataType**: The data type of the variable
- **ValueType**: The type of the observation (field observation, sample, or derived value)
- **SampleMedium**: The sample medium (air, water or other)
- **TimeUnits**: The time units
- **TimeSupport**: The length of the time period of one measurement
Examples

\#Getting all time series from the (14.1E, 49.9N, 14.3E, 50.1N) bounding box
series_catalog <- HISCentral_GetSeriesCatalog(west=14.1, south=49.9, east=14.3, north=50.1)

HISCentral_GetSites

Description

This function gets the table of sites from the HIS Central catalog

Usage

HISCentral_GetSites(west = -180, south = -90, east = 180, north = 90,
serviceID = NULL, keyword = NULL, IncludeServerDetails = TRUE)

Arguments

west
  The west longitude of the geographic bounding box in decimal degrees. Allowed
  values are between -180.0 and +180.0

south
  The south latitude of the geographic bounding box in decimal degrees. Allowed
  values are between -90.0 and +90.0

east
  The east longitude of the geographic bounding box in decimal degrees. Allowed
  values are between -180.0 and +180.0

north
  The north latitude of the geographic bounding box in decimal degrees. Allowed
  values are between -90.0 and +90.0

serviceID
  (optional): The ID of the service on HIS Central. To get the service ID, use the
  id column in the output of the GetServices() function.

keyword
  (optional): The concept keyword (common name of variable) for searching
  the sites on HIS Central. Examples include Temperature, Precipitation, Snow
  Depth,... If the Keyword is not specified then sites with any variable will be
  returned.

IncludeServerDetails
  (optional): If set to TRUE, then the output will include the servCode and servURL
  for each site. If set to FALSE, then we assume that all sites are from the same
  server and the servURL and servCode are not included.

Value

a data.frame of sites. The data.frame has the following columns:

• SiteName: The name of the site
• SiteCode: A short unique code of the site
• FullSiteCode: The complete unique code of the site in the format NETWORK:CODE. Use
  this value in the GetSiteInfo and GetValues functions
• Latitude: The WGS84 latitude in decimal degrees
• Longitude: The WGS84 longitude in decimal degrees
• ServCode: The code of the service in HIS Central. Same as the networkName in the output from GetServices() function. This column is only shown if IncludeServerDetails is TRUE.
• ServURL: The URL of the web service for this site as registered in HIS Central. This column is only shown if IncludeServerDetails is TRUE.

Examples

Getting all sites from the (14.1E, 49.8N, 14.6E, 50.2N) bounding box from the GLDAS web service
sites <- HISCentral_GetSites(west=14.1, south=49.8, east=14.6, north=50.2, serviceID=262)

Description

A helper function that makes a SOAP envelope to send to the CUAHSI WaterOneFlow SOAP web service. It is internally used by the GetSites, GetSiteInfo, GetVariables and GetValues functions.

Usage

MakeSOAPEnvelope(CUAHISINamespace, MethodName, parameters = NULL)

Arguments

CUAHISINamespace
  The SOAP namespace. This must be either "http://www.cuahsi.org/his/1.0/ws" for WaterML 1.0, or "http://www.cuahsi.org/his/1.1/ws" for WaterML 1.1
MethodName
  The name of the WaterOneFlow web service method. It can be one of the following values: "GetSites", "GetSitesObject", "GetSitesByBoxObject", "GetSiteInfoObject", "GetVariablesObject", "GetValuesObject"
parameters
  An optional vector of named parameters for the web method. For GetSites, GetSitesObject and GetVariables no parameters are required. For GetSiteInfoObject you need the "site" parameter. For GetValuesObject you need the "location", "variable", "startDate" and "endDate" parameters.

Value

A <soap:Envelope> text in XML format. This text is send in a HTTP POST body to the SOAP service. Two headers must be sent in the request: Content-Type="text/XML" and SOAPAction=paste(CUAHISINamespace, MethodName). For example if MethodName is GetSites and the WaterML version is 1.1, then SOAPAction="http://www.cuahsi.org/his/1.1/ws/GetSites".
Examples

```r
library(httr)
library(XML)
myenvelope <- MakeSOAPEnvelope("http://www.cuahsi.org/his/1.1/ws/", "GetSitesObject")
SOAPAction <- "http://www.cuahsi.org/his/1.1/ws/GetSitesObject"
url <- "http://hydrodata.info/chmi-d/cuahsi_1_1.asmx"
response <- POST(url, body = myenvelope,
                 add_headers("Content-Type" = "text/xml", "SOAPAction" = SOAPAction),
                 verbose())
status.code <- http_status(response)$category
WaterML <- xmlParse(response)
WaterML
```

WaterMLVersion WaterMLVersion

Description

A helper function that finds out the WaterML version from the WaterML document. By default it checks for "http://www.opengis.net/waterml/2.0" Otherwise it tries to detect "http://www.cuahsi.org/waterML/1.1/" (for WaterML 1.1) or "http://www.cuahsi.org/WaterML/1.0/" (for WaterML 1.0)

Usage

 WaterMLVersion(doc)

Arguments

doc The XML document object

Value

A character with the WaterML version: either 1.0, 1.1, or 2.0

Examples

```r
library(httr)
library(XML)
url <- "http://www.waterml2.org/KiWIS-WML2-Example.wml"
response <- GET(url)
doc <- xmlParse(response)
version <- WaterMLVersion(doc)
```
WaterOneFlowNamespace

Description
A helper function that finds out the WaterOneFlow namespace information based on the version number 1.0 or 1.1.

Usage
WaterOneFlowNamespace(version)

Arguments
version
The version of the WaterOneFlow XML namespace. Must be either "1.0" or "1.1"

Value
A list with the namespaces and corresponding prefixes. This namespace information is important for correct parsing of the WaterML XML document.

Examples
ns <- WaterOneFlowNamespace("1.0")
ns <- WaterOneFlowNamespace("1.1")

WaterOneFlowVersion

Description
A helper function that finds out the WaterOneFlow service version from the URL of the wsdl file. By default it checks for "cuahsi_1_0" or "cuahsi_1_1" in the url. If that is not found, then the function checks the version inside the WSDL file.

Usage
WaterOneFlowVersion(WSDL)

Arguments
WSDL
The URL of the WSDL, for example http://icewater.usu.edu/MudLake/cuahsi_1_0.asmx?WSDL
Value

A list with two items: Version (either 1.0 or 1.1), and Namespace (either http://www.cuahsi.org/His/1.0/ws/ or http://www.cuahsi.org/His/1.1/ws/)

Examples

```r
versionInfo <- WaterOneFlowVersion("http://icewater.usu.edu/MudLake/cuahsi_1_0.asmx?WSDL")
versionInfo$Version
versionInfo$Namespace
```
Index

*Topic **CUAHSI**,
  WaterML-package, 2

*Topic **HydroServer**,
  WaterML-package, 2

*Topic **WaterML**
  WaterML-package, 2

*Topic **WaterML AddValues**, 6
  GetVariables, 13
  MakeSOAPEnvelope, 17
  WaterMLVersion, 18
  WaterOneFlowNamespace, 19
  WaterOneFlowVersion, 19

*Topic **services**
  WaterML-package, 2

*Topic **waterml**
  AddMethods, 3
  AddSites, 4
  AddSources, 5
  AddVariables, 7
  GetServices, 8
  GetSiteInfo, 9
  GetSites, 11
  GetValues, 12
  HISCentral_GetSeriesCatalog, 14
  HISCentral_GetSites, 16

*Topic **web**
  WaterML-package, 2

  AddMethods, 3
  AddSites, 4
  AddSources, 5
  AddValues, 6
  AddVariables, 7

  GetServices, 8
  GetSiteInfo, 9
  GetSites, 11
  GetValues, 12
  GetVariables, 13

HISCentral_GetSeriesCatalog, 14
HISCentral_GetSites, 16
MakeSOAPEnvelope, 17
WaterML-package, 2
WaterMLVersion, 18
WaterOneFlowNamespace, 19
WaterOneFlowVersion, 19