pyWaterML

Release v1.1.24

Elkin Giovanni Romero Bustamante

Aug 04, 2023

MODULES:

PyWaterMLis a python package developed by the Hydroinformatics laboratory in the Civil Engineering at Brigham Young University. PyWaterML allows the user to work with WaterOneFlow (WOF) web services that are compliant with the WaterML 1.0 format. The package lets you connect to any WOF web service such as the 'Consortium of Universities for the Advancement of Hydrological Science, Inc.' ('CUAHSI') WOF web services. PyWaterML strives to be an intuitive python package for accessing WaterML information through WOF web services, and it also brings new functionality to the retrieved data from WOF web services. PyWaterML also brings some basic machine learning clustering based on monthly average for different variables.

WOF Available methods:

1) GetSites(): Get all the sites from a endpoint that complies to the SOAP protocol. The GetSites() method is similar to the GetSites() WaterML method.
2) GetSitesByBoxObject(): Get all the sites from a bounding box from a endpoint that complies to the SOAP protocol. The GetSitesByBoxObject() method is similar to the GetSitesByBoxObject() WaterML method.
3) GetVariables(): Get all the variables from a endpoint that complies to the SOAP protocol. GetVariables() method is similar to the GetVariables() WaterML method.
4) GetSiteInfo(): Get the information of a given site. GetSiteInfo() method is similar to the GetSiteInfo() WaterML method.
5) GetValues(): Get the specific values for an specific variable in a site. GetValues() method is similar to the GetValues() WaterML method.
6) AvailableServices(): Get the WOF web services that are available from a WOF service containing a HIS catalog.
7) GetWaterOneFlowServiceInfo(): Get all registered data services from a given WOF Web service containing a HIS catalog. GetWaterOneFlowServiceInfo can be regarded as a special case of GetServicesInBox2, as the former requests the returns for the global area.

Extra functionality includes the following methods:

1) GetSitesByVariable(): Get the specific sites according to a variable search array from a endpoint that complies to the SOAP protocol. The GetSitesByVariable() is an addition to the WaterML methods because it allows the user to retrieve sites that contains the specific site/s.
2) GetInterpolation(): Interpolates the data given by the GetValues method in order to fix datasets with missing values. Three options for interpolation are offered: mean, backward, forward, default is mean interpolation.
3) GetMonthlyAverage(): Gets the monthly averages for a given variable, or from the response given by the GetValues method for a given site.
4) GetClustersMonthlyAvg(): Gets "n" number of clusters using dtw time series interpolation for a given variable.

The following links will result useful to the users that want to contribute or experiment the package:

1) Source Code
2) Gist Template

CHAPTER

ONE

WATERML MODULE

1.1 Functions to available to Manage WaterOneFlow Web Services

`class pywaterml.waterML.WaterMLOperations(url=None)`

This class represents the WaterML object that will be able to fetch and analyze Data from 'WaterML' and 'WaterOneFlow' Web Services

Parameters
url – WaterOneFlow web service that complies to the SOAP protocol

`AddService(url)`

Add a WaterOneFlow web service to the WaterMLOperations class. It can have any WaterOneFlow web service that uses the SOAP protocol.

Parameters
url – WaterOneFlow web service that complies to the SOAP protocol

Returns
None

Example:

```
url_testing = "http://hydroportal.cuahsi.org/para_la_naturaleza/cuahsi_1_1.asmx?
↪WSDL"
water = WaterMLOperations()
data = water.AddEndpoint(url_testing)
```

`ChangeService(url)`

Change the WaterOneFlow web service of a WaterMLOperations class. The current WaterOneFlow web service can be changed by any WaterOneFlow web service that uses the SOAP protocol.

Parameters
url – WaterOneFlow web service that complies to the SOAP protocol

Returns
None

Example:

```
url_testing = "http://hydroportal.cuahsi.org/para_la_naturaleza/cuahsi_1_1.asmx?
↪WSDL"
water = WaterMLOperations(url = url_testing)
data = water.ChangeEndpoint("http://128.187.106.131/app/index.php/dr/services/
↪cuahsi_1_1.asmx?WSDL")
```

`AvailableServices()`

Give the WaterOneFlow web services that are available from a WaterOneFlow service containing a HIS catalog.

Parameters
`url` – WaterOneFlow web service that complies to the SOAP protocol

Returns
available services in a given WaterOneFlow service containing a HIS catalog.

Return type
hs_services

Example:

```
url_testing = "http://gs-service-production.geodab.eu/gs-service/services/essi/
↪view/whos-country/hiscentral.asmx"
water = WaterMLOperations(url = url_testing)
available_services = water.AvailableServices(url_testing)
```

`GetWaterOneFlowServicesInfo()`

Get all registered data services from a given WaterOneFlow Web service containing a HIS catalog. GetWaterOneFlowServiceInfo can be regarded as a special case of GetServicesInBox2, as the former requests the returns for the global area. :param None:

Returns

- servURL: URL of the WaterOneFlow web service
- Title: title of the WaterOneFlow web service
- organization: supervising organization of the WaterOneFlow web service
- aabstract: abstract of the WaterOneFlow web service

Return type
A dictionary containing the following data for the different WaterOneFlow web services contained in the HIS catalog

Example:

```
url_testing = "http://gs-service-production.geodab.eu/gs-service/services/essi/
↪view/whos-country/hiscentral.asmx"
water = WaterMLOperations(url = url_testing)
services = water.GetWaterOneFlowServiceInfo()
```

`GetSites`(*format='json'*)

Get all the sites from a WaterOneFlow web service that complies to the SOAP protocol. The GetSites() function is similar to the GetSites() WaterML function.

Parameters
`format` – format of the response (json, csv or waterML)

Returns

- latitude = The WGS84 latitude in decimal degrees
- longitude = The WGS84 longitude in decimal degrees
- site_name = The name of the site
- network = Network that the site belongs to

- sitecode = A short unique code of the site
- siteID = The site ID in the original database
- fullSiteCode = full site code of the current site. The fullSiteCode of every site is the following string: "network: sitecode"

Return type

A json, csv or waterML file containing the following data for all the differet sites

Example:

```
url_testing = "http://hydroportal.cuahsi.org/para_la_naturaleza/cuahsi_1_1.asmx?
↪WSDL"
water = WaterMLOperations(url = url_testing)
sites = water.GetSites()
```

GetSitesByBoxObject(*ext_list*, *inProjection*, *format='json'*)

Get all the sites from a bounding box from a WaterOneFlow web service that complies to the SOAP protocol. The GetSitesByBoxObject() function is similar to the GetSitesByBoxObject() WaterML function.

Parameters

- **ext_list** – Array of bounding box coordinates in a given projection.
- **inProjection** – Projection from the array of coordinates of the given bounding box.
- **format** – format of the response (json, csv or waterML)

Returns

A json, csv or waterML file containing the following data for all the differet sites in the selected boundingbox

- latitude = The WGS84 latitude in decimal degrees
- longitude = The WGS84 longitude in decimal degrees
- site_name = The name of the site
- network = Network that the site belongs to
- sitecode = A short unique code of the site
- siteID = The site ID in the original database
- fullSiteCode = full site code of the current site. The fullSiteCode of every site is the following string: "network: sitecode"

Example:

```
url_testing = "http://hydroportal.cuahsi.org/para_la_naturaleza/cuahsi_1_1.asmx?
↪WSDL"
water = WaterMLOperations(url = url_testing)
## use with epsg:4326 ##
BoundsRearranged = [-66.4903,18.19699,-66.28665,18.28559]
sites = water.GetSitesByBoxObject(BoundsRearranged,'epsg:4326')
```

GetVariables(*format='json'*)

Get variables meatada from a WaterOneFlow web service that complies to the SOAP protocol. GetVariables() function is similar to the GetVariables() WaterML function

Parameters

format – format of the response (json, csv or waterML)

Returns

- variableName: Name of the variable
- unitName: Name of the units of the values associated to the given variable and site
- unitAbbreviation: unit abbreviation of the units from the values associated to the given variable and site
- noDataValue: value associated to lack of data.
- isRegular: Boolean to indicate whether the observation measurements and collections regular
- timeSupport: Boolean to indicate whether the values support time
- timeUnitName: Time Units associated to the observation
- timeUnitAbbreviation: Time units abbreviation
- sampleMedium: the sample medium, for example water, atmosphere, soil.
- speciation: The chemical sample speciation (as nitrogen, as phosphorus..)

Return type

A json, csv or waterML file containing the following data of the variables from the WaterOneFlow web service

Example::

url_testing = "http://hydroportal.cuahsi.org/para_la_naturaleza/cuahsi_1_1.asmx?WSDL" water = WaterMLOperations(url = url_testing) variables = water.GetVariables()

`GetSiteInfo`(*site_full_code*, *format='json'*)

Get the information of a given site. GetSiteInfo() function is similar to the GetSiteInfo() WaterML function.

Parameters

- **site_full_code** – A string representing the full code of the given site following the structure - site_full_code = site network + ":" + site code
- **format** – format of the response (json, csv or waterML)

Returns

- siteName: Name of the site.
- siteCode: Code of the site.
- network: observation network that the site belongs to
- fullVariableCode: The full variable code, for example: SNOTEL:SNWD.Use this value as the variableCode parameter in GetValues().
- siteID: ID of the site
- latitude: latitude of the site
- longitude: longitude of the site
- variableName: Name of the variable
- unitName: Name of the units of the values associated to the given variable and site
- unitAbbreviation: unit abbreviation of the units from the values associated to the given variable and site
- dataType: Type of data

- noDataValue: value associated to lack of data.
- isRegular: Boolean to indicate whether the observation measurements and collections regular
- timeSupport: Boolean to indicate whether the values support time
- timeUnitName: Time Units associated to the observation
- timeUnitAbbreviation: Time units abbreviation
- sampleMedium: the sample medium, for example water, atmosphere, soil.
- speciation: The chemical sample speciation (as nitrogen, as phosphorus..)
- beginningDateTimeUTC: The UTC date and time of the first available
- EndDateTimeUTC: The UTC date and time of the last available
- beginningDateTime: The local date and time of the first available
- EndDateTime: The local date and time of the last available
- 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)
- methodCode: The code of the method or instrument used for the observation
- methodID: The ID of the sensor or measurement method
- qualityControlLevelCode: The code of the quality control level. Possible values are -9999(Unknown), 0 (Raw data), 1 (Quality controlled data), 2 (Derived products), 3 (Interpretedproducts), 4 (Knowledge products)
- qualityControlLevelID: The ID of the quality control level. Usually 0 means raw data and 1 means quality controlled data.
- sourceCode: The code of the data source.
- timeOffSet: The difference between local time and UTC time in hours.

Return type

A json, csv or waterML file containing the following data of the seleceted site from the WaterOneFlow web service

Example:

```
url_testing = "http://hydroportal.cuahsi.org/para_la_naturaleza/cuahsi_1_1.asmx?
↪WSDL"
water = WaterMLOperations(url = url_testing)
sites = water.GetSites()
firstSiteFullSiteCode = sites[0]['fullSiteCode']
siteInfo = water.GetSiteInfo(firstSiteFullSiteCode)
```

GetValues(*site_full_code*, *variable_full_code*, *start_date*, *end_date*, *methodCode=None*, *qualityControlLevelCode=None*, *format='json'*)

Get the specific values for an specific variable in a site. GetValues() function is similar to the GetValues() WaterML function.

Parameters

- **site_full_code** – A string representing the full code of the given site following the structure - site_full_code = site network + ":" + site code

- `variable_full_code` – A string representing the full code of the given variable following the structure - variable_full_code = site network + ":" + variable code
- `start_date` – beginning date time for the time series of the variable
- `end_date` – end date time for the time series of the variable
- `methodCode` – method code for data extraction for the given variable
- `qualityControlLevelCode` – The ID of the quality control level.Typically 0 is used for raw dataand 1 is used for quality controlled data.To get a list of possible quality controllevel IDs, see qualityControlLevelCode column in the output of GetSiteInfo(). If qualityControlLevelCode is not specified,then the observations in the output data.frame won't befiltered by quality control level code.
- `format` – format of the response (json, csv or waterML)

Returns

- siteName: Name of the site.
- siteCode: Code of the site.
- network: observation network that the site belongs to
- siteID: ID of the site
- latitude: latitude of the site
- longitude: longitude of the site
- variableName: Name of the variable
- unitName: Name of the units of the values associated to the given variable and site
- unitAbbreviation: unit abbreviation of the units from the values associated to the given variable and site
- dataType: Type of data
- noDataValue: value associated to lack of data.
- isRegular: Boolean to indicate whether the observation measurements and collections regular
- timeSupport: Boolean to indicate whether the values support time
- timeUnitName: Time Units associated to the observation
- timeUnitAbbreviation: Time units abbreviation
- sampleMedium: the sample medium, for example water, atmosphere, soil.
- speciation: The chemical sample speciation (as nitrogen, as phosphorus..)
- dateTimeUTC: The UTC time of the observation.
- dateTime: The local date/time of the observation.
- dataValue: Data value from the observation.
- 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)
- methodCode: The code of the method or instrument used for the observation

- qualityControlLevelCode: The code of the quality control level. Possible values are -9999(Unknown), 0 (Raw data), 1 (Quality controlled data), 2 (Derived products), 3 (Interpretedproducts), 4 (Knowledge products)
- sourceCode: The code of the data source
- timeOffSet: The difference between local time and UTC time in hours.

Return type
An object containing properties for the time series values for the given variable in the given site. The object has the following data

Example:

```
url_testing = "http://hydroportal.cuahsi.org/para_la_naturaleza/cuahsi_1_1.asmx?
↪WSDL"
water = WaterMLOperations(url = url_testing)
sites = water.GetSites()
firstSiteFullSiteCode = sites[0]['fullSiteCode']
siteInfo = water.GetSiteInfo(firstSiteFullSiteCode)
firstVariableFullCode = siteInfo['siteInfo'][0]['fullVariableCode']
start_date = siteInfo['siteInfo'][0]['beginDateTime'].split('T')[0]
end_date = siteInfo['siteInfo'][0]['endDateTime'].split('T')[0]
variableResponse= water.GetValues(site_full_code, variable_full_code, start_
↪date, end_date)
```

GetSitesByVariable(*specific_variables_codes*, *cookiCutter=None*, *format='json'*)

Get the specific sites according to a variable search array from a WaterOneFlow web service that complies to the SOAP protocol. The GetSitesByVariable() is an addition to the WaterML functions because it allows the user to retrieve sites that contains the epecific site/s.

Args

specific_variables: An array of strings representing a list of variables that will serve as a filter when retrieving sites. cookiCutter: A list containing the different information from each site. It can be the response of the GetSites() or GetSitesByBoxObject() functions. if the cookiCutter is not specified, the function will filter all the functions calling GetSites() internally. format: format of the response (json, csv or waterML)

Returns

- latitude = The WGS84 latitude in decimal degrees
- longitude = The WGS84 longitude in decimal degrees
- site_name = The name of the site
- network = Network that the site belongs to
- sitecode = A short unique code of the site
- siteID = The site ID in the original database
- fullSiteCode = full site code of the current site. The fullSiteCode of every site is the following string: "network: sitecode"

Return type
An array of objects that represent each site. The structure of the response is the following

Example:

```
url_testing = "http://hydroportal.cuahsi.org/para_la_naturaleza/cuahsi_1_1.asmx?
↪WSDL"
water = WaterMLOperations(url = url_testing)
sites = water.GetSites()['sites']
variables = water.GetVariables()['variables']

# choose the first variable to filter#

variables_to_filter = [variables[0][variableCode]]
sitesFiltered = water.GetSitesByVariable(variables_to_filter,sites)
```

GetInterpolation(*GetValuesResponse*, *type='mean'*)

Interpolates the data given by the GetValues function in order to fix datasets with missing values. Three ooptions for interpolation are offered: mean, backward, forward. The default is the mean interpolation.

Parameters

- **GetValuesResponse** – response from the GetValues() function
- **type** – type of interpolation to be performed: mean, backward, forward
- **format** – format of the response (json, csv or waterML)

Returns

An array containing the interpolation chosen by the user (backward, mean, forward)

Example:

```
url_testing = "http://hydroportal.cuahsi.org/para_la_naturaleza/cuahsi_1_1.asmx?
↪WSDL"
water = WaterMLOperations(url = url_testing)
sites = water.GetSites()
firstSiteFullSiteCode = sites[0]['fullSiteCode']
siteInfo = water.GetSiteInfo(firstSiteFullSiteCode)
firstVariableFullCode = siteInfo['siteInfo'][0]['fullVariableCode']
start_date = siteInfo['siteInfo'][0]['beginDateTime'].split('T')[0]
end_date = siteInfo['siteInfo'][0]['endDateTime'].split('T')[0]
variableResponse= water.GetValues(site_full_code, variable_full_code, start_
↪date, end_date)
interpolationData = water.GetInterpolation(variableResponse, 'mean')
```

GetMonthlyAverage(*GetValuesResponse=None*, *site_full_code=None*, *variable_full_code=None*, *start_date=None*, *end_date=None*, *methodCode=None*, *qualityControlLevelCode=None*)

Gets the monthly averages for a given variable, or from the response given by the GetValues function for a given site.

Parameters

- **GetValuesResponse** – response from the GetValues() function. If this is given the others paramters do not need to be given.
- **site_full_code** – A string representing the full code of the given site following the structure - site_full_code = site network + ":" + site code
- **variable_full_code** – A string representing the full code of the given variable following the structure - variable_full_code = site network + ":" + variable code
- **start_date** – beginning date time for the time series of the variable

- **end_date** – end date time for the time series of the variable
- **methodCode** – method code for data extraction for the given variable
- **qualityControlLevelCode** – The ID of the quality control level.Typically 0 is used for raw dataand 1 is used for quality controlled data. To get a list of possible quality controllevel IDs, see qualityControlLevelCode column in the output of GetSiteInfo(). If qualityControlLevelCode is not specified,then the observations in the output data.frame won't befiltered by quality control level code.

Returns

An object containing properties for the time series values for the given variable in the given site. The structure of the response is the following

- variable: variable name
- unit: units of the values
- title: title of the time series values
- values: an array of arrays containing [date, value]

Example:

```
water = WaterMLOperations(url = url_testing)
sites = water.GetSites()
firstSiteFullSiteCode = sites[0]['fullSiteCode']
siteInfo = water.GetSiteInfo(firstSiteFullSiteCode)
firstVariableFullCode = siteInfo['siteInfo'][0]['fullVariableCode']
start_date = siteInfo['siteInfo'][0]['beginDateTime'].split('T')[0]
end_date = siteInfo['siteInfo'][0]['endDateTime'].split('T')[0]
variableResponse= water.GetValues(site_full_code, variable_full_code, start_
↪date, end_date)
monthly_averages = water.getMonthlyAverage(variableResponse)
```

GetClustersMonthlyAvg(*sites*, *variableCode*, *n_cluster=3*, *methodCode=None*, *qualityControlLevelCode=None*, *timeUTC=False*)

Gets "n" number of clusters using dtw time series interpolation for a given variable

Parameters

- **sites** – response from the GetSites() function. Performance of the fuction can be given if the resuls of the GetSitesByVariable() function is passed instead.
- **variableCode** – string representing the variable code for the time series clusters of the given sites.
- **n_clusters** – integer representing the number of cluster to form.
- **methodCode** – method code for data extraction for the given variable.
- **qualityControlLevelCode** – The ID of the quality control level.Typically 0 is used for raw dataand 1 is used for quality controlled data. To get a list of possible quality controllevel IDs, see qualityControlLevelCode column in the output of GetSiteInfo(). If qualityControlLevelCode is not specified, then the observations in the output data.frame won't befiltered by quality control level code.
- **timeUTC** – Boolean to use the UTC time instead of the time of the observation.

Returns

An array of arrays of the following structure [monthly averages array, cluster_id]

[[[0.141875, 0.1249375, 0.0795, 0.12725, 0.0877, 0.0, 0.09375, 0.1815, 0.15437499999999998, 0.164625, 0.1614, 0.20900000000000002], 1], [[0.1, 0.08662500000000001, 0.0414025, 0.048, 0.052, 0.0, 0.1105, 0.015, 0.06625, 0.10587500000000001, 0.0505, 0.046125], 0], [[0.2265, 0.27225, 0.17407499999999998, 0.13475, 0.14525, 0.129, 0.17825, 0.210625, 0.103125, 0.0, 0.23675], 2]]

Example:

```
url_testing = "http://hydroportal.cuahsi.org/para_la_naturaleza/cuahsi_1_1.asmx?
↪WSDL"
water = WaterMLOperations(url = url_testing)
sites = water.GetSites()
firstSiteFullSiteCode = sites[0]['fullSiteCode']
siteInfo = water.GetSiteInfo(firstSiteFullSiteCode)['siteInfo']
clusters = water.getClustersMonthlyAvg(sites,siteInfo[0]['variableCode'])
```

CHAPTER

TWO

WATERANALITYCA MODULE

2.1 Helper Functions used in the WaterML module to analyze data

`class pywaterml.analyzeData.WaterAnalityca`

This class represents the Analitics object for the WaterMLOperations class. The WaterAnalityca provides functions related to statistics(monthly and daily averages) and basic Math functionality(Interpolation). However, this is a helper class for the main WaterMLOperations class.

`_Interpolate`(*type='mean'*, *timeUTC=False*)

Helper function to rerieve different kinds of interpolation in the WaterMLOperations GetInterpolation() function :param GetValuesResponse: GetValues Response from WaterMLOperations GetValues() function :param type: Type of interpolation

Returns
Interpolated data

Return type
dataInterpolated

`_MonthlyAverages`()

Helper function to rerieve the monthly averages from the WaterMLOperations GetValues() function in the GetMonthlyAverage() function :param GetValuesResponse: GetValues Response from WaterMLOperations GetValues() function

Returns
monthly_average array

Return type
m_avg

`_DailyAverages`()

Helper function to rerieve the daily averages from the WaterMLOperations GetValues() function in the GetDailyAverage() function :param GetValuesResponse: GetValues Response from WaterMLOperations GetValues() function

Returns
daily_average array

Return type
d_avg

CHAPTER

THREE

AUXILIARY MODULE

3.1 Helper Functions used in the WaterML module

class pywaterml.auxiliaryMod.**GetSoapsPlugin**

This class represents the GetSoapsPlugin object for the WaterMLOperations class. The GetSoapsPlugin provides two functions of the MessagePlugin: the reveived and sending functions. It helps for debugging purposesrealted to the SOAP protocol.

Parameters

MessagePlugin – The MessagePlugin currently has (5) hooks: - marshalled():: Provides the plugin with the opportunity to inspect/modify the envelope Document before it is sent. - sending():: Provides the plugin with the opportunity to inspect/modify the message text before it is sent. - received():: Provides the plugin with the opportunity to inspect/modify the received XML text before it is SAX parsed. - parsed():: Provides the plugin with the opportunity to inspect/modify the sax parsed DOM tree for the reply before it is unmarshalled. - unmarshalled():: Provides the plugin with the opportunity to inspect/modify the unmarshalled reply before it is returned to the caller.

sending(*context*)

Provides the plugin with the opportunity to inspect/modify the message text before it is sent. Args: The Context classes which are passed to the plugin. :returns: None

received(*context*)

Provides the plugin with the opportunity to inspect/modify the received XML text before it is SAX parsed. Args: The Context classes which are passed to the plugin. :returns: None

class pywaterml.auxiliaryMod.**Auxiliary**

This class represents the Auxiliary object for the WaterMLOperations class. The Auxiliary provides functions related to parsing data from XML to JSON format, and the creation of JSON responses to have a more digested output. However, this is a helper class for the main WaterMLOperations class.

_parseJSON()

Helper function to parse JSON data into a python dictionary. It is used in the WaterMLOperations GetSites() function. :param json: json object

Returns

Dictionary from all the sites of an specific URL with the following data:

- latitude = The WGS84 latitude in decimal degrees
- longitude = The WGS84 longitude in decimal degrees
- site_name = The name of the site

- network = Network that the site belongs to
- sitecode = A short unique code of the site
- siteID = The site ID in the original database
- fullSiteCode = full site code of the current site. The fullSiteCode of every site is the following string: "network: sitecode"

Return type
hs_sites

`_parseWML`(*bbox*)

Helper function to parse JSON data from a bounding box into a python dictionary . It is used in the WaterMLOperations GetSitesByBoxObject() function. :param bbox: json object from belonging to the bounding box

Returns

Dictionary from all the sites of an specific WaterOneFlow web service with the following data:

- latitude = The WGS84 latitude in decimal degrees
- longitude = The WGS84 longitude in decimal degrees
- site_name = The name of the site
- network = Network that the site belongs to
- sitecode = A short unique code of the site
- siteID = The site ID in the original database
- fullSiteCode = full site code of the current site. The fullSiteCode of every site is the following string: "network: sitecode"

Return type
hs_sites

`_recursive_asdict`(*d*)

Helper function to make Suds object into serializable format recurvesively . It is used in the _parseJSON and _parseWML functions. :param d: json object

Returns
None

`_getValuesHelper`(*k*, *return_obj*)

Helper function to parse and store the content of the dictionary response from the GetValues at the level (['timeSeriesResponse']['timeSeries']['values']['value']) into a new dictionary. The data stored into this dictionary from the GetValues response is the following:

- dateTimeUTC: The UTC time of the observation.
- dateTime: The local date/time of the observation.
- dataValue: Data value from the observation.
- 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)
- methodCode: The code of the method or instrument used for the observation
- qualityControlLevelCode: The code of the quality control level. Possible values are -9999(Unknown), 0 (Raw data), 1 (Quality controlled data), 2 (Derived products), 3 (Interpretedproducts), 4 (Knowledge products)

- sourceCode: The code of the data source
- timeOffSet: The difference between local time and UTC time in hours.

This function is only stores half of the reponse from the GetValues method, and it is usually used with the _getValuesHelper2 function that stores the other half of the function. :param k: GetValues response dictionary at level -> ([‘timeSeriesResponse’][‘timeSeries’][‘values’][‘value’]) :param return_obj: python dictionary that will store the data from teh GetValues response.

Returns
python dictionary containing data from the GetValues response.

Return type
return_obj

`_getValuesHelper2`(*times_series*, *return_object*)

Helper function to parse and store the content of the dictionary response from the GetValues at the level ([‘timeSeriesResponse’][‘timeSeries’][‘values’][‘value’]) into a new dictionary. The data stored into this dictionary from the GetValues response is the following:

- siteName: Name of the site.
- siteCode: Code of the site.
- network: observation network that the site belongs to
- siteID: ID of the site
- latitude: latitude of the site
- longitude: longitude of the site
- variableName: Name of the variable
- unitName: Name of the units of the values associated to the given variable and site
- unitAbbreviation: unit abbreviation of the units from the values associated to the given variable and site
- dataType: Type of data
- noDataValue: value associated to lack of data.
- isRegular: Boolean to indicate whether the observation measurements and collections regular
- timeSupport: Boolean to indicate whether the values support time
- timeUnitName: Time Units associated to the observation
- timeUnitAbbreviation: Time units abbreviation
- sampleMedium: the sample medium, for example water, atmosphere, soil.
- speciation: The chemical sample speciation (as nitrogen, as phosphorus..)

This function is only stores half of the reponse from the GetValues method, and it is usually used with the _getValuesHelper function that stores the other half of the function. :param times_series: GetValues response dictionary at level -> ([‘timeSeriesResponse’][‘timeSeries’][‘values’][‘value’]) :param return_object: python dictionary that will store the data from teh GetValues response.

Returns
python dictionary containing data from the GetValues response.

Return type
return_object

`_getSiteInfoHelper`(*object_siteInfo*, *object_methods*)

Helper function to parse and store the content of two dictionaries:

- object_methods = GetSiteInfoResponse ['sitesResponse']['site']['seriesCatalog']['series']
- object_siteInfo = GetSiteInfoResponse ['sitesResponse']['site']['siteInfo']

Both dictionaries containing the response from the GetSiteInfo at store the following content into a new dictionary:

- siteName: Name of the site.
- siteCode: Code of the site.
- network: observation network that the site belongs to
- fullVariableCode: The full variable code, for example: SNOTEL:SNWD.Use this value as the variableCode parameter in GetValues().
- siteID: ID of the site
- latitude: latitude of the site
- longitude: longitude of the site
- variableName: Name of the variable
- unitName: Name of the units of the values associated to the given variable and site
- unitAbbreviation: unit abbreviation of the units from the values associated to the given variable and site
- dataType: Type of data
- noDataValue: value associated to lack of data.
- isRegular: Boolean to indicate whether the observation measurements and collections regular
- timeSupport: Boolean to indicate whether the values support time
- timeUnitName: Time Units associated to the observation
- timeUnitAbbreviation: Time units abbreviation
- sampleMedium: the sample medium, for example water, atmosphere, soil.
- speciation: The chemical sample speciation (as nitrogen, as phosphorus..)
- beginningDateTimeUTC: The UTC date and time of the first available
- EndDateTimeUTC: The UTC date and time of the last available
- beginningDateTime: The local date and time of the first available
- EndDateTime: The local date and time of the last available
- 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)
- methodCode: The code of the method or instrument used for the observation
- methodID: The ID of the sensor or measurement method
- qualityControlLevelCode: The code of the quality control level. Possible values are -9999(Unknown), 0 (Raw data), 1 (Quality controlled data), 2 (Derived products), 3 (Interpretedproducts), 4 (Knowledge products)
- qualityControlLevelID: The ID of the quality control level. Usually 0 means raw data and 1 means quality controlled data.

- sourceCode: The code of the data source.
- timeOffSet: The difference between local time and UTC time in hours.

Parameters

- `object_siteInfo` – Contains metadata associated to the site.
- `object_methods` – Contains a list of <series>, which are unique combinations of site, variable and time intervals that specify a sequence of observations.

Returns
python dictionary containing data from the GetSiteInfo response.

Return type
return_obj

`_getVariablesHelper`(*one_variable*, *return_object*)

Helper function to parse and store the content of the GetValues response dictionary at the level:

- one_variable = GetVariablesResponse [‘variablesResponse’][‘variables’][‘variable’]

The dictionary containing the response from the GetValues method stores the following content into a new dictionary:

- variableName: Name of the variable
- unitName: Name of the units of the values associated to the given variable and site
- unitAbbreviation: unit abbreviation of the units from the values associated to the given variable and site
- noDataValue: value associated to lack of data.
- isRegular: Boolean to indicate whether the observation measurements and collections regular
- timeSupport: Boolean to indicate whether the values support time
- timeUnitName: Time Units associated to the observation
- timeUnitAbbreviation: Time units abbreviation
- sampleMedium: the sample medium, for example water, atmosphere, soil.
- speciation: The chemical sample speciation (as nitrogen, as phosphorus..)

Parameters

- `one_variable` – Contains metadata associated to the different variables of the site.
- `return_object` – python dictionary that will store the data from the GetVariables response.

Returns
python dictionary containing data from the GetVariables response.

Return type
return_object

`_parseService`(*centralUrl*)

Helper function to parse JSON data into a python dictionary. It is used in the WaterMLOperations GetWaterOneFlowServiceInfo() function. If the WaterOneFlow web service endpoint is not accesible though the suds library. :param centralUrl: URL from a WaterOneFlow web servicee to access a HIS catalog

Returns

Dictionary from all the web services contained in the WaterOneFlow web service endpoint. The folllowing data is returned for each service:

- servURL: URL of the WaterOneFlow web service
- Title: title of the WaterOneFlow web service
- organization: supervising organization of the WaterOneFlow web service
- aabstract: abstract of the WaterOneFlow web service

Return type
services

`_giveServices`(*services*, *filter_serv=None*)

Helper function to retrieve the WaterOneFlow web services of a catalog as a python dictionary. It is used in the WaterMLOperations GetWaterOneFlowServiceInfo() function to parse the different services. :param services: Array of URL WaterOneFlow web services. :param filter_serv: Filter to only parse some services that meet a condition.

Returns

Dictionary from all the web services contained in the WaterOneFlow web service endpoint. The folllowing data is returned for each service:

- url: URL of the WaterOneFlow web service
- title: title of the WaterOneFlow web service
- description: abstract of the WaterOneFlow web service

Return type
json_response

CHAPTER

FOUR

BYU HYDROINFORMATICS LABORATORY

4.1 About us

The Hydroinformatics laboratory at BYU focuses in delivering different software product and services for water modeling. Some of the most important work involves: Global streamflow Forecast Services API, creation of the Tethys Platform, Hydroserver_lite, and much more. The most recent publication and work can be found in official BYU_Hydroinformatics website

PYTHON MODULE INDEX

INDEX