IBG-3 SOS command line-client (clisos.py)

Juergen Sorg and Ralf Kunkel

Forschungszentrum Jülich GmbH
Institute für Bio- und Geowissenschaften
Agrospäre (IBG-3)
D-524525 Jülich

E-Mail {j.sorg|r.kunkel}@fz-juelich.de
Telefon: +49-2461-61 {5535|3262}

Dokument-Nr: IBG3-2019-01
Version: 1.1
Datum: 2019-02-01

1 Introduction

The program clisos.py is a command line client for access to the data provided by the Institute for Bio- and Geosciences (IBG-3) of Forschungszentrum Jülich GmbH via standardized OGC Sensor Observation Services (SOS). For more detailed information on the SOS standard, see [1]. The client described here accesses these services via the Hypertext Transfer Protocol (HTTP), allowing access independent of the internal structure of the data.

2 Execution of clisos

The clisos.py program is implemented in the Python programming language and requires a Python interpreter to execute. The program itself is platform-independent, as long as a Python interpreter is installed on the respective system (Linux, Windows, MacOX). Free versions of Python can be downloaded at http://www.python.org/. clisos was developed and tested with Python version 2.7.3.
The program execution is made with the command:

python clisos.py *OPTION*

OPTION	Definition
-h	hostname/servername (e.g. www.menja.de)
-p	port (e.g. 8080)
-u	url/path (e.g. /ibg3sosV1.0/sos
-P	get the entire capabilities document from the service
-o	specify the offering to use
-O	Get a list of all available Offerings
-s	specify the station to use
-Y	get all stations (procedures)
-S	get all stations (procedures) from an offering
-D	get all parameters (observedProperties) from a station
-L	get all parameters (observedProperties) from a Offering
-X	print station description in xml
-G	get data by a GetObservation-Request output is csv like with comma separated values timestamp,featureOfInterest,param0,...,paramN (exactly output format depends on sos-output) output will be written to stdout, therefore use stdout forwarding (... > filename) to save output to a file
-f	read/write from/to a configuration file
-C	create a configuration file for a certain offering

[1] https://www.opengeospatial.org/standards/sos

-r	spatial reference system to use for entire Request default is urn:ogc:def:crs:EPSG::4326
-c	spatial reference system for spatial filter
-x	x-coordinate of Spatial Filter defined by a Point
-y	y-coordinate of Spatial Filter defined by a Point
-q	proxy hostname
-w	proxy port
-j	specify the responseFormat paramter of a SOS GetObservation request
-z	use this character to separate the data values within csv file
-l	plots the data
-a	comma separated format (color and symbol) string for the plotting curve e.g. "g.,r-" the first curve is plotted with green (g) dots (.) and the second with red (r) lines (-) (see http://matplotlib.org/1.3.1/api/pyplot_api.html#matplotlib.pyplot.plot for more possible format parameters)
-b	specify a comma separated list of observed properties
-d	specify a comma separated list of procedures
-e	specify start and end time stamp separated with semicolon for data retrieva e.g. 2005-10-13T04:03:14,2007-09-20T16:04:10
-g	specify a spatial bbox filter. syntax: xmin,ymin,xmax,ymax
-i	no output when no data is available (else a message about this is returned)
-k	print no header information (print only data)
-m	print result om document (no parsing, no csv)
-n	get latest/first data record (0==latest,1==first)
-t	only count result rows
-B	providing Bearer-Token

3 Explanations

3.1 Show Help

Specifying the *--help* option on the command line displays all available command line options with a brief description:

python clisos.py --help

3.2 Getting Information about the requested service

The http address of a Sensor Observation Service has the structure:

http://teodoor.icg.kfa-juelich.de:80/eifelrur_public_oauth/sos

and therefore consists of three components:

1. Hostname: *teodoor.icg.kfa-juelich.de*
2. Port: *80*
3. Path to the service: */eifelrur_public_oauth/sos*

accordingly, a call to clisos.py to request this service would require the following information:

python clisos.py –h teodoor.icg.kfa-juelich.de –p 80 –u /eifelrur_public_oauth/sos

3.3 Offerings

The data provided by the SOS are organized in so-called offerings. These offerings may be either groups of parameters (e.g., climate, soil) or groups of measurement sites (e.g., Wüstebach, Rur). An offering generally includes several stations and parameters.

In order to retrieve data from the service, it must be determined, if not already known, which offerings are provided via the service. This is done via the Option –O:

python clisos.py -h *host* -p *port* -u *path* -O

If the command line option -O is passed, clisos.py determines the offerings provided by the SOS and writes them to the standard output.

3.4 Mesaurementstations

The option -S can be used to determine all stations (measuring points) that are provided by the SOS in an offering. The offering must be specified with the -o option:

python clisos.py -h *host* -p *port* -u *path* -o *offering* –S

3.5 Parameter

There are two ways to get information about the provided parameters.

1. In the first case one is interested in obtaining a list of all parameters of a particular measuring point. The option -D is available for this purpose:

   python clisos.py -h *host* -p *port* -u *path* -s *station* –D

2. In the second case, one would like to obtain a list of all parameters published (regardless of metering) via an offering. The option -L can be used for this:

   python clisos.py -h *host* -p *port* -u *path* -d *offering* -L

In both cases, the program creates a listing and outputs it to the console.

3.6 Description of station

If a precise description of a specific measuring point (for example name or coordinates) is required, this can be requested via the -X option. The result is encoded in XML (SensorML) and output on the console:

python clisos.py -h *host* -p *port* -u *path* -s *station* -X

4 Download observational data

The options described so far enable the information supplied by an SOS to be queried. In order to retrieve the actual observational data from the SOS, the measuring points, parameters, offerings and also the desired period of time must be known. This information must be provided via a configuration file.

4.1 Configuration file for data download

Configuration file consists of two sections:

- *getObservation*-Section (starting with *[getObservation]*) and
- a *connection*-Section, starting with *[connection]*

The respective sections consist of property-value pairs in the form:

Property = Value0 [, Value1]

Separator is a comma. Depending on the property, either no value, exactly one value or any number of values can be specified. In the following table, the optional (cardinality ≥ 0) or mandatory (cardinality ≥ 1) parameters are specified for the *getObservation* section. For clarity, Appendix 1 shows an example of a configuration file.

Property	**Description**	**Cardinality**
offering	Offering	1
stations	Comma separated list of station names	0..n
parameter	Comma separated list of parameter names (ObservedProperties)	0..n
spatialfilterpoint	Coordinates of point for spatial filter as Y X	0..1
crs	Spatial references system of filter	0..1
resultmodel	Resulting data model; either om:Observation or om:TimeSeriesObservation.	1
starttime	Start timestamp for the desired time interval Format: YYYY-MM-DDThh:mm:ss	1
endtime	End timestamp for the desired time interval Format: YYYY-MM-DDThh:mm:ss	1

The connection section contains all information about the requested service and has the form:

[connection]
host = *host*
port = *port*
url = *path*

4.2 Generating a configuration file

To avoid unnecessary paperwork, clisos.py provides the ability to create a startup file that contains all the metrics and parameters of an offering. This is done through the options *-C* und *–f.*

python clisos.py -h *host* -p *port* -u *path* -f *file* –o *offering* -C

When using the -C option, the name of the configuration file (file) to be created with the -f file option and the desired offering (offering) must also be specified with the option -o offering.

The created file can be modified individually with a text editor, while retaining the basic structure of the file (see Appendix 1)..

4.3 Data download

Data retrieval is initiated via the options -G and -f, where -f specifies the name of the configuration file (file) to be used:

python clisos.py -f *file* –G

Note that host, port, and path do not need to be specified here because they are already specified in the configuration file. After calling the program, the data is output in a comma-separated list on the standard output.

Hint: Since all stations and parameters of an offering are included in the generation of the configuration file, it may happen that data is not available for all parameters for a station. In these cases, the content of the data is inconsistent with the parameters specified in the header. It must therefore be checked by the user, if such a case has occurred. If necessary, in these cases, the data must be downloaded through several configuration files.

4.4 Redirecting the standard output to a file

If you want to output the downloaded data to a file, you can do so with the command

python clisos.py -f *file* -G > *outputfile*

This redirects the data, but also any error messages, to the output file. If you want to output the data sorted by date, then following command can be used

python clisos.py -f *file* –G | sort > *outputfile*

Appendix 1: Sample configuration file

```
[getObservation]
offering = Public
stations = SE_EC_001
parameter = SoilHeatFlux_0.02mAvg10min, SoilTemperature_0.01mAvg10min,
AirPressure_1m_Avg10min,  Radiation_Global_Avg10min, WindSpeed_2m_Avg30min,
WindSpeed_2m_Avg10min, AirZOverObukhovLength, AirHeatFlux_Sensible_RelErrNoise,
AirConcentration_CO2_2m_Avg30min, WindDirection_2mAvg30min,
Radiation_PhotoActive_Avg10min,  AirHeatFlux_Sensible_2m_Avg30min,
AirFlux_CO2_Avg30min_BelowObsHeight, Radiation_ShortWaveOut_Avg10min,
AirHumidity_Absolute_2m_Avg30min_LiCor, AirHumidity_Absolute_2m_Avg10min,
FootprintAdajcentAreaContribution_Cum30Min, AirHeatFlux_Latent_2m_Avg30min,
Flux_CO2Flux_Avg30min, Radiation_Net_Avg10min, SoilWaterContent_0.02mAvg10minSensor2,
Precipitation_Cum10min_TB,  SoilTemperature_0.05mAvg10min,
FootprintSourceContribution_Cum30Min, AirHumidity_Absolute_2m_Avg10min,
AirHumidity_Absolute_2m_Avg30min_HMP, FootprintDistanceSourceContribution_Max,
SoilWaterContent_0.02mAvg10minSensor1,  AirTemperature_2m_Avg30min,
AirTemperature_2m_Avg10min_Sensor2, AirTemperature_2m_Avg10min_Sensor1,
SoilHeatFlux_0.08mAvg10minSensor2, SoilHeatFlux_0.08mAvg10minSensor1,
AirVaporPressure_2mAvg10min, SoilTemperature_0.04mAvg10min, WindFrictionVelocity,
Radiation_LongWaveOut_Avg10min,Radiation_LongWaveIn_Avg10min
resultmodel = om:Observation
starttime = 2010-12-31T22:00:00
endtime = 2019-01-01T02:20:00

[connection]
host = teodoor.icg.kfa-juelich.de
port = 80
url = /eifelrur_public_oauth/sos
```