# Package ‘sos4R’

April 29, 2020

<table>
<thead>
<tr>
<th>Type</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Client for OGC Sensor Observation Services</td>
</tr>
<tr>
<td>Version</td>
<td>0.4.0</td>
</tr>
<tr>
<td>Date</td>
<td>2020-04-22</td>
</tr>
<tr>
<td>Depends</td>
<td>R (&gt;= 3.4.0)</td>
</tr>
<tr>
<td>Imports</td>
<td>dplyr, httr, methods, parsedate, sp, stats, stringr, uuid, xml2 (&gt;= 1.2.2),</td>
</tr>
<tr>
<td>Suggests</td>
<td>readr, spacetime, leaflet, leafpop, mapdata, maps, maptools, mapview, xts, testthat, rgdal, knitr, rmarkdown, webmockr, kableExtra, skimr, clipr</td>
</tr>
<tr>
<td>Description</td>
<td>A client for Sensor Observation Services (SOS, see <a href="https://www.opengeospatial.org/standards/sos">https://www.opengeospatial.org/standards/sos</a>) as specified by the Open Geospatial Consortium (OGC). With the package users can retrieve (meta)data from SOS instances and interactively create requests for near real-time observation data based on the available sensors, phenomena, observations etc. using thematic, temporal, and spatial filtering.</td>
</tr>
<tr>
<td>License</td>
<td>GPL-2</td>
</tr>
<tr>
<td>URL</td>
<td><a href="https://github.com/52North/sos4R">https://github.com/52North/sos4R</a></td>
</tr>
<tr>
<td>Encoding</td>
<td>UTF-8</td>
</tr>
<tr>
<td>LazyLoad</td>
<td>TRUE</td>
</tr>
<tr>
<td>ByteCompile</td>
<td>TRUE</td>
</tr>
<tr>
<td>BugReports</td>
<td><a href="https://github.com/52North/sos4R/issues">https://github.com/52North/sos4R/issues</a></td>
</tr>
<tr>
<td>VignetteBuilder</td>
<td>knitr</td>
</tr>
</tbody>
</table>
R topics documented:

SOS-methods-accessor.R PrintShowStructureSummary-methods.R
SOS-methods-coercion.R SML-methods-util.R
SOS_200-methods-parsing.R SOS_200_methods-gda.R
Development.R

RoxygenNote 7.1.0
Language en-GB
NeedsCompilation no

Author Daniel Nuest [cre, aut] (<https://orcid.org/0000-0002-0024-5046>),
  Edzer Pebesma [ctb] (<https://orcid.org/0000-0001-8049-7069>),
  Ben Graeler [aut] (<https://orcid.org/0000-0001-5443-4304>),
  Benjamin Pross [aut],
  Eike Hinderk Juerrens [aut],
  52°North Initiative for Geospatial Open Source Software GmbH [cph]

Maintainer Daniel Nuest <daniel.nuest@uni-muenster.de>
Repository CRAN

Date/Publication 2020-04-29 15:00:02 UTC

R topics documented:

sos4R-package ............................................................... 3
Coercion ................................................................. 5
Constants ............................................................... 6
Defaults ................................................................. 7
DescribeSensor ......................................................... 10
encodeKVP-methods .................................................. 13
encodeRequestKVP-methods ......................................... 13
encodeRequestSOAP-methods ....................................... 14
encodeRequestXML-methods ........................................... 14
encodeXML-methods .................................................. 15
getCapabilities-methods ............................................ 16
data ................................................................. 17
getFeatureOfInterest .................................................. 18
GetObservation ......................................................... 19
GML ................................................................. 23
MonitoringPoint-class ............................................... 29
OGC ................................................................. 30
OmMeasurement ......................................................... 32
OmObservation-class .................................................. 34
OmObservationCollection ............................................ 36
OmOM_Observation-class ............................................. 37
sos4R-package

A client for the OGC Sensor Observation Service

Description

sos4R is a client for Sensor Observation Services (SOS). It allows users to retrieve metadata from SOS web service instances as specified by the Open Geospatial Consortium (OGC) and subsequently to interactively create requests for observation data based on the available sensors, phenomena, observations, offerings etc.

Details

Package: sos4R
Type: Package
Version: 0.4.0
Date: 2020-04-21
License: GPL-2
LazyLoad: yes
ByteCompile: yes
Imports: xml2, htr, sp, stringr, methods

Note
The development of this software was gratefully supported by the 52North Student Innovation Prize for Geoinformatics 2010.
To stay updated on all matters around sos4R go to the development blog at http://www.nordholmen.net/sos4r/.
If you want to ask questions about using the software, please go to the issue tracker at https://github.com/52North/sos4R/issues.
The most extensive documentation is contained in the package vignettes.

Author(s)
Daniel Nuest <daniel.nuest@uni-muenster.de>

References

See Also
See also the package vignette.

Examples

## Not run:

# Open the connection
sos = SOS(url = "<the service endpoint>")

# List offerings, procedures and observedProperties
names(sosOfferings(sos))
sosProcedures(sos)
sosObservedProperties(sos)

# Create time period (last 30 days)
tPeriod <- sosCreateEventTimeList(
time = sosCreateTimePeriod(
sos = pegelsos,
begin = Sys.time() - (3600 * 24 * 30),
end = Sys.time()))
Coercion

# Request data for all observed properties and procedures of a certain offering
observation <- getObservation(sos = sos,
observedProperty = sosObservedProperties(sos),
offering = sosOfferings(sos)[[2]],
procedure = sosProcedures(sos),
eventTime = tPeriod)

# Inspect result
sosResult(observation)
utils::str(sosResult(observation))

# Inspect attributes of the data fields
if(is.list(sosResult(observation))) {
  attributes(sosResult(observation)[,1])
} else {
  attributes(sosResult(pegelObs)[,1])
}

# Use custom converting function and connection method. This mechanism works the
# same for encoders and decoders.
myConverters <- SosDataFieldConvertingFunctions(
  "myNumericUnit" = sosConvertDouble)
mySos <- SOS(sos.url, binding = "KVP", dataFieldConverters = myConverters)
sosDataFieldConverters(mySos)

# where to find the cheat sheet
sosCheatSheet()

# global objects caching objects during parsing of XML documents
cacheVariables

## End(Not run)

---

Coercion

Coercion of SOS objects to Spatial classes

Description

The package sos4R comes with several classes that contain spatial information, such as coordinates of sensors or observation data associated with coordinates. For easy access to these coordinates, there are coercion functions to matching classes from the package sp.

Details

Most coercions, e.g. from Gml.. classes, are used internally during parsing and result extraction. Objects containing data, e.g. OmObservationCollection, return a SpatialPointsDataFrame. Objects not containing data, e.g. SensorML instances, return an object of class SpatialPoints. To just get the most fitting class, coerce to "Spatial" as shown in the examples.
Examples

# sensor location
mySOS <- SOS(url = "http://sensorweb.demo.52north.org/sensorwebtestbed/service/kvp",
             binding = "KVP")
mySensor <- describeSensor(sos = mySOS,
                           procedure = sosProcedures(mySOS)[[1]],
                           outputFormat = "text/xml; subtype="sensorML/1.0.1"")
as(mySensor, "Spatial")

# offering bounding box
as(sosOfferings(mySOS)[[1]], "Spatial")

---

Constants in sos4R

Description

The package sos4R comes with a set of constant character strings and fixed supported features, for example for names of XML elements, XML Namespace prefixes, or supported formats and models.

Details

Most of these variables should be pretty self-explanatory.

Constants for names of XML elements start with a lowercase character string of the namespace prefix (e.g. "gml"), a unique name of the element (where parts like "type" and special characters may be left out, and other descriptive elements may be added for clarity), and end with "Name".

Examples: codegmlEnvelopeName, ogcGeometryOperandLineStringName, ogcTempOpTMEqualsName.

The OwsExceptionsData() function provides access to the fixed exception codes, meanings and respective HTTP codes and messages.

The mime types are used for automatic detection of the best fitting parser.

References

See also Defaults for default parameter settings.

OGC 06-121r3, Version: 1.1.0 with Corrigendum 1

Examples

# example constants
sos100NamespacePrefix
gmlNameName
sweUomName

# Data frame holding OWS exception code information
OwsExceptionsData()
# Get all namespaces
SosAllNamespaces("1.0.0")

---

**Defaults**

**Default Parameter Settings and Handling Functions**

---

**Description**

These values are default parameters and handling functions for connections and requests to, as well as response processing of answers from, Sensor Observation Services. These allow to simplify a SOS connection for the most common use cases and non-expert users.

**Usage**

SosDefaultBinding()

SosParsingFunctions(..., include = character(0), exclude = character(0))

SosEncodingFunctions(..., include = character(0), exclude = character(0))

SosDataFieldConvertingFunctions(..., include = character(0), exclude = character(0))

SosDisabledParsers()

SosDefaults()

SosResetParsingFunctions(sos)

SosDefaultDCPs()

SosDefaultParsingOptions()

**Arguments**

... Named references to functions to be used for the respective element during parsing, encoding or conversion, e.g. "myUnit" = myUnitParser.

include A list of names of elements whose functions shall be included in the returned list, e.g. include = c("GetObservation","DescribeSensor"). This inclusion is done after replacing the default functions based on the ... argument.

exclude A list of names of elements whose functions shall be excluded in the returned list, e.g. exclude = c("DescribeSensor"). This exclusion is done after replacing the default functions based on the ... argument.

sos An object of class SOS.
Details
The default values are strongly related to what is actually implemented in the package, but also often resemble the (hopefully) most common use cases.

Some defaults are accessed directly, others should be accessed using a function. The latter is required for cases where a runtime evaluation is needed, e.g. for default values of construction functions.

A special case are the functions to access the default functions for specific purposes, which are the parsing functions, the encoding functions and the field converting functions. See the examples on how to use them.

The function `SosDisabledParsers` can be used to use no parsing at all (despite the parsing for the capabilities response, which is required for establishing a connection to a SOS. This function is helpful to inspect the unprocessed responses from a service.

The function `SosResetParsingFunctions` can be used to replace the included parsing functions of a SOS object with the default ones. This is even useful for development of the default parsing functions.

The default parameter values are:

- `sosDefaultCharacterEncoding`  
  `{sosDefaultCharacterEncoding}`
- `sosDefaultDescribeSensorOutputFormat`  
  `{sosDefaultDescribeSensorOutputFormat}`
- `sosDefaultGetCapSections`  
  `{sosDefaultGetCapSections}`
- `sosDefaultGetCapAcceptFormats`  
  `{sosDefaultGetCapAcceptFormats}`
- `sosDefaultGetCapOwsVersion`  
  `{sosDefaultGetCapOwsVersion}`
- `sosDefaultGetObsResponseFormat`  
  `{sosDefaultGetObsResponseFormat}`
- `sosDefaultTimeFormat`  
  `{sosDefaultTimeFormat}`
- `sosDefaultFilenameTimeFormat`  
  `{sosDefaultFilenameTimeFormat}`
- `sosDefaultTempOpPropertyName`  
  `{sosDefaultTempOpPropertyName}`
- `sosDefaultSpatialOpPropertyName`  
  `{sosDefaultSpatialOpPropertyName}`

The default parsing functions can be replaced for a variety of XML elements, so that you only need to replace the parts of the parsing that really must be changed. Be aware that inclusion and exclusion are performed after merging the given functions with the defaults!

Example Services: This list contains a few SOS instances that were tested (to different degrees) with sos4R. The package authors do not maintain these services, so no guarantee can be given that these are usable.

Value
The default value of the respective setting or parameter. This can be a list, especially a named list of functions.

References
- Constants
Examples

# simple default values
show(sosDefaultCharacterEncoding)
show(sosDefaultDescribeSensorOutputFormat)
show(sosDefaultGetCapAcceptFormats)
show(sosDefaultGetCapOwsVersion)
show(sosDefaultGetCapSections)
show(sosDefaultGetObsResponseFormat)
show(sosDefaultSpatialOpPropertyName)
show(sosDefaultTempOpPropertyName)
show(sosDefaultTemporalOperator)
show(sosDefaultTimeFormat)
SosDefaultBinding()

## Not run:
# usage of defaults in construction method for SOS class
sos <- SOS("http://mysos.com/sos", binding = SosDefaultBinding(),
timeFormat = sosDefaultTimeFormat)

# functions to disable all parsing
SosDisabledParsers()

## End(Not run)

# Replace a parsing function
myER <- function(xml) {
  return("EXCEPTION!!!11")
}
myParsingFunctions <- SosParsingFunctions("ExceptionReport" = myER)

# use inclusion and exclusion, important: even the just added function needs to
# be included manually!
myParsingFunctions <- SosParsingFunctions("ExceptionReport" = myER,
  include = c("GetObservation", "DescribeSensor", "ExceptionReport"))

myParsingFunctions <- SosParsingFunctions(exclude = c("GetObservation", "DescribeSensor"))

## Not run:
# Replace an encoding function
myEncoding <- function(object, v) {
  return(utils::str(object))
}

sos = SOS(url = "http://mysos.com/sos",
  encoders = SosEncodingFunctions("POST" = myPostEncoding))

# Use custom converting function and connection method. This mechanism works the
# same for encoders and decoders.
myConverters <- SosDataFieldConvertingFunctions("myNumericUnit" = sosConvertDouble)
mySos <- SOS(sos.url, binding = "KVP", dataFieldConverters = myConverters)
sosDataFieldConverters(mySos)

# inspecting XML using dummy parsing function
sos = SOS(url = "http://mysos.com/sos", parsers = SosDisabledParsers)
describeSensor(sos, sosProcedures(sos)[[1]])

# replace the parsing functions with the default ones
sos <- SosResetParsingFunctions(sos)

## End(Not run)

# a named list of all defaults
SosDefaults()

# parsing options used by xml2
SosDefaultParsingOptions()

---

**DescribeSensor**

*Retrieve Sensor Descriptions from a SOS*

**Description**

The DescribeSensor operation of a Sensor Observation Service can be used to retrieve metadata of procedures that are available from a SOS. This sensor description is normally encoded in SensorML. The method `describeSensor(...)` sends a DescribeSensor request for a description of the given procedure to the given Sensor Observation Service instance. If the procedure is a vector, the method will send multiple requests and combine the results in a list.

Please also consult the specification for details on possible contents of the request.

This functions should not be called directly, but instead using the function `describeSensor`.

**Usage**

```r
describeSensor(sos, procedure, outputFormat, validTime, verbose, inspect, saveOriginal)
```

# internal constructors:
# exactly one of outputFormat or procedureDescriptionFormat must be used
SosDescribeSensor(service = NA_character_,
                  version = NA_character_,
                  procedure = NA_character_,
                  outputFormat = NA_character_,
                  procedureDescriptionFormat = NA_character_,
                  validTime = NULL)

**Arguments**

- **service**
  - The service attribute of the request, e.g. ‘SOS’.
- **sos**
  - The SOS connection to use.
- **version**
  - The version attribute of the request, e.g. ‘1.0.0’.
- **procedure**
  - The value of the procedure elements in the request, e.g. ‘urn:procedure:42’.
outputFormat The value of the output format element in the request, e.g. 'text/xml; subtype="sensorML/1.0.1"'.

procedureDescriptionFormat
The desired description format for the procedure, e.g. 'http://www.opengis.net/sensorML/1.0.1'. For using facing function describeSensor(..) the value of outputFormat is used.

validTime An object of class GmlTimeObject, either a time instant or time period for which the returned description shall be valid; use functions sosCreateTimePeriod(..) and sosCreateTimeInstant(..) to create this parameter.

verbose A boolean value indicating whether debug information is printed out to the console during the execution.

inspect A boolean value to enable printing of the sent request and received response to the console.

saveOriginal Save the received document to the current working directory. If TRUE a filename is automatically generated, if a character string is given it is used as the filename.

Value
The value of the construction function is an object of class SosDescribeSensor-class.

Objects from the Class
Objects can be created by calling the construction function of the form DescribeSensor(...). The contain the procedure identifier that is to be described by a service.

Methods
signature(sos = "SOS", procedure = "character") Method requests a description of the given procedure(s) from the given SOS.

Slots
procedure: Object of class "character", the identifier of the procedure.
outputFormat: Object of class "character", the requested output format.
procedureDescriptionFormat: Object of class "character", the requested description format.
validTime: Object of class "GmlTimeObject", the time instant or period for which the description shall be valid.

service: Object of class "character", the service type, e.g. "SOS".
request: Object of class "character", the name of the request, "DescribeSensor".
version: Object of class "character", the service version, e.g. "1.0.0"

Extends
Class "OwsServiceOperation", directly.
Methods

- **encodeRequestKVP** signature(obj = "DescribeSensor"): Encode the information in the request as key-value-pairs for HTTP GET connections, see [encodeRequestKVP-methods](#).

- **encodeRequestSOAP** signature(obj = "DescribeSensor"): Encode the information in the request as XML for SOAP connections, see [encodeRequestSOAP-methods](#).

- **encodeRequestXML** signature(obj = "DescribeSensor"): Encode the information in the request as XML for HTTP POST connections, see [encodeRequestXML-methods](#).

- **show** signature(object = "DescribeSensor"): Show a human readable excerpt of the contents of the object.

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

References

See OGC 06-009r6 section 8.4, or the XSD schema file at [http://schemas.opengis.net/sos/1.0.0/sosDescribeSensor.xsd](http://schemas.opengis.net/sos/1.0.0/sosDescribeSensor.xsd).

See Also

See Also [SensorML](#) and [describeSensor](#).

Examples

```r
showClass("SosDescribeSensor")

# example for construction function
describeSensorRequest <- SosDescribeSensor(service = "SOS", version = "1.0.0", procedure = "urn:procedure:42", outputFormat = "text/xml")
print(describeSensorRequest)

# encode the request in XML
sos <- SOS_Test()
encodeRequestXML(describeSensorRequest, sos)
toString(encodeRequestXML(describeSensorRequest, sos))

# request a sensor description
mySOS <- SOS(url = "http://sensorweb.demo.52north.org/sensorwebtestbed/service/kvp",
             binding = "KVP")
mySensor <- describeSensor(sos = mySOS, procedure = sosProcedures(mySOS)[[1]],
                           outputFormat = 'text/xml; subtype="sensorML/1.0.1"', # space is needed!
                           )
```
**Encode Classes as KVP**

**Description**

These methods convert a given object to a key-value-pair representation to be used in GET requests. The given instance of SOS is possibly used for encoding sub-elements or accessing metadata which is required for the encoding, like time stamp format.

**Methods**

- `signature(obj = "GmlTimeInstant", sos = "SOS")` Convert the given object to a KVP representation.
- `signature(obj = "GmlTimePeriod", sos = "SOS")` Convert the given object to a KVP representation.
- `signature(obj = "GmlTimePosition", sos = "SOS")` Convert the given object to a KVP representation.
- `signature(obj = "OgcBinaryTemporalOp", sos = "SOS")` Convert the given object to a KVP representation.
- `signature(obj = "SosEventTime", sos = "ANY")` Convert the given object to a KVP representation.
- `signature(obj = "POSIXt", sos = "ANY")` Convert the given object to a string format suitable for KVP representation.

**See Also**

[SosBindings](https://en.wikipedia.org/wiki/Key_Value_Pair)

---

**Methods for Encoding Requests to SOS in KVP Format**

**Description**

These methods encode objects representing requests to a Sensor Observation Service into a key-value-pair format which can be used in the GET binding, see SosBindings.

**Methods**

- `signature(obj = "SosDescribeSensor")` Encode a DescribeSensor request.
- `signature(obj = "SosGetObservation")` Encode a GetObservation request.
- `signature(obj = "SosGetObservationById")` Encode GetObservationById request.
signature(obj = "OwsGetCapabilities") Dispatching method, checks the version attribute and forwards the encoding to the appropriate method. This method should be called rather than calling the versioned methods directly!

signature(obj = "OwsGetCapabilities_1.1.0") Encode GetCapabilities request with OWS version 1.1.0.

signature(obj = "OwsGetCapabilities_2.0.0") Encode GetCapabilities request with OWS version 2.0.0.

encodeRequestSOAP-methods

Methods for Encoding Requests to SOS in SOAP Format

Description

These methods encode objects representing requests to a Sensor Observation Service into a SOAP message format to be used in the SOAP binding (see SosBindings).

Methods

signature(obj = "SosDescribeSensor") Encode a DescribeSensor operation.

signature(obj = "SosGetObservation") Encode a GetObservation operation.

signature(obj = "SosGetObservationById") Encode a GetObservationById operation.

signature(obj = "OwsGetCapabilities") Encode a GetCapabilities operation.

See Also

SosBindings, encodeXML

encodeRequestXML-methods

Methods for Encoding Requests to SOS in XML Format

Description

These methods encode objects representing requests to a Sensor Observation Service into a XML format which can be used in the POST binding, see SosBindings.

Methods

signature(obj = "SosDescribeSensor") Encode a DescribeSensor request.

signature(obj = "SosGetObservation") Encode a GetObservation request.

signature(obj = "SosGetObservationById") Encode a GetObservationById request.

signature(obj = "OwsGetCapabilities") Encode a GetCapabilities request.
encodeXML-methods

References

See Also
SosBindings, encodeXML

---

**Description**
These methods convert the given objects to XML representations for HTTP POST requests. The given instance of SOS is possibly used for encoding sub-elements or accessing metadata which is required for the encoding, like time stamp format.

**Methods**

- `signature(obj = "GmlDirectPosition", sos = "SOS")` Convert the given object to an XML representation.
- `signature(obj = "GmlEnvelope", sos = "SOS")` Convert the given object to an XML representation.
- `signature(obj = "GmlLineString", sos = "SOS")` Convert the given object to an XML representation.
- `signature(obj = "GmlPointProperty", sos = "SOS")` Convert the given object to an XML representation.
- `signature(obj = "GmlPoint", sos = "SOS")` Convert the given object to an XML representation.
- `signature(obj = "GmlPolygon", sos = "SOS")` Convert the given object to an XML representation.
- `signature(obj = "GmlTimeInstantProperty", sos = "SOS")` Convert the given object to an XML representation.
- `signature(obj = "GmlTimeInstant", sos = "SOS")` Convert the given object to an XML representation.
- `signature(obj = "GmlTimePeriod", sos = "SOS")` Convert the given object to an XML representation.
- `signature(obj = "GmlTimePosition", sos = "SOS")` Convert the given object to an XML representation.
- `signature(obj = "POSIXt", sos = "SOS")` Converts the time object to a string that is suitable to be used as the value of XML time elements.
- `signature(obj = "OgcBBOX", sos = "SOS")` Convert the given object to an XML representation.
- `signature(obj = "OgcComparisonOps", sos = "SOS")` Convert the given object to an XML representation.
signature(obj = "OgcContains", sos = "SOS") Convert the given object to an XML representation.
signature(obj = "OgcIntersects", sos = "SOS") Convert the given object to an XML representation.
signature(obj = "OgcOverlaps", sos = "SOS") Convert the given object to an XML representation.
signature(obj = "SosEventTime", sos = "SOS") Convert the given object to an XML representation.
signature(obj = "SosFeatureOfInterest", sos = "SOS") Convert the given object to an XML representation.
signature(obj = "SosGetFeatureOfInterest_2.0.0", sos = "SOS") Convert the given object to an XML representation.
signature(obj = "TM_After", sos = "SOS") Convert the given object to an XML representation.
signature(obj = "TM_Before", sos = "SOS") Convert the given object to an XML representation.
signature(obj = "TM_During", sos = "SOS") Convert the given object to an XML representation.
signature(obj = "TM_Equals", sos = "SOS") Convert the given object to an XML representation.
signature(obj = "XMLNode", sos = "SOS") Convert the given object to an XML representation.

References

XML specification: http://www.w3.org/XML/.

See Also

encodeRequestXML, encodeRequestSOAP

---

**getCapabilities-methods**

*Request Capabilities from a SOS*

**Description**

This method request the metadata description of a given Sensor Observation Service, the Capabilities document.

**Methods**

- signature(sos = "SOS") Request capabilities description from a SOS.
- signature(inspect = "logical") Print out the sent and received documents to console.
- signature( verbose = "logical") Extensive debugging information printed to console.
Description
For given sites and phenomena, a SOS is queried and the data returned as data.frame.

Usage
getData(sos, phenomena, sites, spatialBBox = NA, begin = NA, end = NA, ...)

Arguments
- **sos**: An SOS object from a call to `SOS` containing the URL of a SOS v2.0.
- **phenomena**: A list, vector or one-column data.frame with characters identifying the relevant phenomena.
- **sites**: A list, vector or one-column data.frame with characters identifying the relevant sites.
- **spatialBBox**: A 2-by-2 matrix with x and y in the rows and min and max in the columns. See `Spatial` for details. If `sites` is set, this argument is ignored.
- **begin**: Object of class `POSIXt`. Note that creating timestamps with `as.POSIXct(..)` will use your local timezone.
- **end**: Object of class `POSIXt`. Note that creating timestamps with `as.POSIXct(..)` will use your local timezone.
- **...**: Named arguments forwarded to other wrapper functions internally.

Value
A data.frame containing the data in long form where each row contains a siteID, the time stamp of the observation and the observed measurements for each phenomenon in a separate column. The measurements are provided with units using `units`.

Author(s)
Benedikt Graeler, Eike Hinderk Juerrens

Examples
```r
## Not run:
mySos <- SOS(url = "https://climate-sos.niwa.co.nz/",
             binding = "KVP", useDCPs = FALSE, version = "2.0.0")

observationData <- getData(sos = mySos, 
                            phenomena = phenomena[18,1],
                            sites = siteList[1:2,1])

## End(Not run)
```
**getFeatureOfInterest**  
*Function retrieving features of interest, i.e. the representations of the real world features that are observed and for which observations are provided.*

---

**Description**

This function retrieves, i.e. the representations of the real world features that are observed and for which observations are provided, from a Sensor Observation Service.

**Usage**

```r
getFeatureOfInterest(sos, featureOfInterest, verbose = sos@verboseOutput, inspect = FALSE, saveOriginal = FALSE)
```

**Arguments**

- **sos**: The Sensor Observation Service from which features of interest should be retrieved.
- **featureOfInterest**: Identifier(s) of features of interest.
- **verbose**: A boolean value indicating whether debug information is printed out to the console during the execution.
- **inspect**: A boolean value to enable printing of the sent request and received response to the console.
- **saveOriginal**: Save the received document to the current working directory. If TRUE a filename is automatically generated, if a character string is given it is used as the filename.

**Value**

An object of class `SosGetFeatureOfInterest_2.0.0-class`.

**Examples**

```r
# Should be DIRECTLY executable !! ----
#-- Getting package 'os' from HTTP --
#This is the comment section
```
GetObservation

Request data with GetObservation and GetObservationById requests

Description

getObservation(...) can be used to retrieve data from a SOS instance by sending a GetObservation request to the given SOS.

getObservationById(...) can be used to retrieve data from a SOS instance by sending a GetObservationById request to the given SOS.

These methods take a variety of inputs (see slot definitions), of which only offering is mandatory for GetObservation operation, and the observationId for GetObservationById operation.

This document also describes classes (and their construction functions) for the operations.

Usage

SosGetObservation(service, version, offering, observedProperty, responseFormat, srsName = as.character(NA), eventTime = list(), procedure = as.character(NA), featureOfInterest = NULL, result = NULL, resultModel = as.character(NA), responseMode = as.character(NA), BBOX = as.character(NA), valueReferenceTemporalFilter = as.character(NA))

SosGetObservationById(service, version, observationId, responseFormat, srsName = as.character(NA), resultModel = as.character(NA), responseMode = as.character(NA))

Arguments

service The service attribute of the request, e.g. ‘SOS’.
version The version attribute of the request, e.g. ‘1.0.0’.
observationId The value of the ObservationId element in the request, e.g. ‘o_12345’, which is to be obtained. This could have been obtained by the client via a URL in a feed, alert, or some other notification.
offering The offering element value in the request, e.g. ‘“temperatures”’. All other parameters are depending on the selected offering. This must be a character string for an object of class SosObservationOffering-class obtained by sosOfferings(...); can be a list for SOS 2.0.0 with both character strings and objects.
observeredProperty A list of values for observedProperty elements in the request, e.g. ‘urn:property:AirTemperature’. IDs of phenomena are advertised in capabilities document.
responseFormat The responseFormat element value in the request, e.g. ‘text/xml; subtype=“om/1.0.0”’. ID of the output format to be used for the requested data. The supported output formats are listed in the selected offering capabilities.
srsName The srsName attribute of the request, e.g. ‘urn:ogc:def:crs:EPSG:4326’.
GetObservation

**eventTime**
A list of objects of class SosEventTime-class which are added as eventTime elements to the request. Allows a client to request observations from a specific instant, multiple instances or periods of time in the past, present and future. The supported range is listed in the selected offering capabilities.

**procedure**
A list of procedure identifiers added to the request as procedure elements.

**featureOfInterest**
An object of class SosFeatureOfInterest added to the request as the featureOfInterest element, or NULL. Specifies target feature for which observations are requested.

**result**
An object of class OgcComparisonOps-class added to the request as result element, or NULL, or any element that can be encoded using encodeXML(...) and then be added to an XML document with addChildren(...). Filtering: Only report observations where the result matches this expression.

**resultModel**
The resultModel element of the request, e.g. 'om:Measurement', which is an identifier of the result model to be used for the requested data. The resultModel values supported by a service are listed in the contents section of the service metadata, identified as QName values.

**responseMode**
The responseMode element of the request, e.g. 'inline', which allows the client to request the form of the response.

**BBOX**
A bounding box to be used only with KVP encoding in request via HTTP GET, in the format 'minlon,minlat,maxlon,maxlat,srsURI?', with the spatial reference system being optional. This element is ignored for POST requests, use the parameter featureOfInterest instead, see SosBindings.

**valueReferenceTemporalFilter**
The property name used in a temporal filter for SOS 2.0 KVP requests, ignore for SOS 1.0.0.

**Details**
Please consult the specification for details on possible contents of the request.

**Value**

**Constructor functions:** An object of class SosGetObservation-class or SosGetObservationById-class respectively.

**Request functions:** A list of or an object of class OmObservationCollection-class or OmObservation-class depending on the returned content.

**Methods**

signature(sos="SOS", offering="SosObservationOffering") or signature(sos="SOS", offering="character")
Request observation data from the given SOS for the given offering (either character identifier or an object of class SosObservationOffering).

signature(sos="SOS", observationId="character")
Request observation data from the given SOS for the given observation identifier.
Objects from the Class

Objects can be created by calls to the construction functions of the form SosGetObservationById(...) or SosGetObservationById(...).

Slots

BBOX: Object of class "character", specifies a bounding box for spatial filtering to be applied in GET requests (only), see SosBindings.

eventTime: Object of class "list", specifies the time period(s) for which observations are requested.

featureOfInterest: Object of class "SosFeatureOfInterestOrNULL", specifies the feature for which observations are requested. This can either be represented by a reference to a feature ID advertised in the capabilities document or can be a spatial constraint.

observationId: Object of class "character", the Id of the requested observation.

observedProperty: Object of class "list", specifies the phenomenon or phenomena for which observations are requested.

oxFor: For SOS 1.0.0: object of class "character" for SOS 2.0.0 a list, specifies the offering advertised in the GetCapabilities document.

procedure: Object of class "list", procedure parameter specifies the sensor system(s) for which observations are requested.

request: Object of class "character", the name of the request.

responseFormat: Object of class "character", specifies the desired resultFormat MIME content type for transport of the results.

responseMode: Object of class "character", specifies whether results are requested in-line, out-of-band, as an attachment, or if this is a request for an observation template that will be used for subsequent calls to GetResult.

resultModel: Object of class "character", specifies the QName of the root element of an O&M Observation or element in the appropriate substitution group.

result: Object of class "ANY", provides a place to put in OGC filter expressions based on property values. This instructs the SOS to only return observations where the result matches this expression.

service: Object of class "character", service type identifier.

srsName: Object of class "character", defines the spatial reference system that should be used for any geometries that are returned in the response. This must be one of the advertised values in the offering specified in gml:srsName elements.

valueReferenceTemporalFilter: Object of class "character", the property name used in a temporal filter for SOS 2.0 KVP requests.

version: Object of class "character", specification version for operation.

Extends

SosGetObservation: Class "OwsServiceOperation", directly.

SosGetObservationById: Class "OwsServiceOperation", directly.
Author(s)
Daniel Nuest <daniel.nuest@uni-muenster.de>

References
See OGC 06-009r6 section 8.4, or the XSD schema file at http://schemas.opengis.net/sos/1.0.0/sosGetObservation.xsd.
See OGC 06-009r6 section 10.1, or the XSD schema file at http://schemas.opengis.net/sos/1.0.0/sosGetObservationById.xsd.

See Also
SosGetObservation-class, SosGetObservationById-class

Examples
## Not run:
# request observations
mySOSpox <- SOS(url = "http://sensorweb.demo.52north.org/sensorwebtestbed/service/pox",
                binding = "POX", useDCPs = FALSE)
myOffering <- sosOfferings(mySOS)[["ws2500"]]
period <- sosCreateTimePeriod(sos = mySOS,
                              begin = as.POSIXct("2015/11/01"),
                              end = as.POSIXct("2015/11/02"))
eventTime <- sosCreateEventTimeList(period)
nov2015 <- getObservation(sos = mySOSpox,
                           offering = myOffering,
                           eventTime = eventTime)
# request observation by identifier and get the data
obsId <- getObservationById(sos = mySOSpox,
                            observationId = "http://www.52north.org/test/observation/1")
sosResult(obsId, coordinates = TRUE)
## End(Not run)

showClass("SosGetObservation")
showClass("SosGetObservationById")
responseFormat <- "text/xml; subtype="om/1.0.0"
obsReq <- SosGetObservation(service = "SOS", version = "1.0.0",
                            offering = "temperatures",
                            observedProperty = list("urn:property:AirTemperature"),
                            responseFormat = responseFormat)
print(obsReq)
obsWithIdReq <- SosGetObservationById(service = "SOS", version = "1.0.0",
                                       observationId = "o_12345",
                                       ...)
Classes and Construction Functions from the GML Namespace

Description

Classes for GML elements and their respective construction functions. See the referenced specification for details.

Usage

GmlDirectPosition(pos, srsName = as.character(NA), srsDimension = NA_integer_, axisLabels = as.character(NA), uomLabels = as.character(NA))
GmlDirectPositionLatLon(lat, lon, srsName = as.character(NA), srsDimension = NA_integer_, axisLabels = as.character(NA), uomLabels = as.character(NA))
GmlEnvelope(lowerCorner, upperCorner, srsName = as.character(NA), srsDimension = NA_integer_, axisLabels = as.character(NA), uomLabels = as.character(NA))
GmlFeatureCollection(featureMembers, id = as.character(NA))
GmlPoint(pos, id = as.character(NA), srsName = as.character(NA), srsDimension = NA_integer_, axisLabels = as.character(NA), uomLabels = as.character(NA))
GmlPointProperty(href = as.character(NA), point = NULL)
GmlFeatureProperty(href = as.character(NA), feature = NULL)
GmlTimeInstant(timePosition, id = as.character(NA), relatedTimes = list(NA), frame = as.character(NA))
GmlTimeInstantProperty(href = as.character(NA), time = NULL)
GmlTimeInterval(interval, unit, radix = NA, factor = NA)
GmlTimePeriod(begin = NULL, beginPosition = NULL, end = NULL, endPosition = NULL, duration = as.character(NA), timeInterval = NULL, id = as.character(NA), relatedTimes = list(NA), frame = as.character(NA))
GmlTimePosition(time, frame = as.character(NA), calendarEraName = as.character(NA), indeterminatePosition = as.character(NA))
GmlMeasure(value, uom)
Arguments

axisLabels  See corresponding slot description.
begin      See corresponding slot description.
beginPosition See corresponding slot description.
calendarEraName See corresponding slot description.
duration  See corresponding slot description.
end        See corresponding slot description.
endPosition See corresponding slot description.
factor     See corresponding slot description.
feature    See corresponding slot description.
featureMembers See corresponding slot description.
frame      See corresponding slot description.
href       See corresponding slot description.
id         See corresponding slot description.
indeterminatePosition See corresponding slot description.
interval   See corresponding slot description.
lat        Latitude coordinate parameter.
lon        Longitude coordinate parameter.
lowerCorner See corresponding slot description.
point      See corresponding slot description.
pos       See corresponding slot description.
radix      See corresponding slot description.
relatedTimes See corresponding slot description.
srsDimension See corresponding slot description.
srsName    See corresponding slot description.
time       See corresponding slot description.
timeInterval See corresponding slot description.
timePosition See corresponding slot description.
unit       See corresponding slot description.
uomLabels  See corresponding slot description.
upperCorner See corresponding slot description.
value      See slot description.
uom        See slot description.

Details

The "...OrNULL" classes are used to model optional slots.
Value

The construction functions return an object of the respective class.

Objects from this classes

Objects can be created by calling the according construction functions. e.g. in the form GmlPoint(...).

Extends

GmlFeature: Class "GmlFeatureOrNULL", directly.
GmlFeatureCollection: Class "GmlFeature", directly. Class "GmlFeatureOrNULL", by class "GmlFeature", distance 2.
GmlLineString: Class "GmlGeometry", directly.
GmlPoint: Class "GmlGeometry", directly.
GmlTimeInterval: Class "GmlTimeIntervalOrNULL", directly.
GmlTimeObject: Class "GmlTimeObjectOrNULL", directly.
GmlTimePrimitive: Class "GmlTimeObject", directly. Class "GmlTimeObjectOrNULL", by class "GmlTimeObject", distance 2.

Virtual Classes

No objects may be created from the following virtual classes: GmlFeature, GmlTimeObject, GmlTimePrimitive, GmlTimeGeometricPrimitive, GmlGeometry, all ...OrNULL classes.

Slots

axisLabels: Object of class "character": An character vector of labels for all the axes of this CRS.
begin: Object of class "GmlTimeInstantPropertyOrNULL": An object of class GmlTimeInstantProperty-class, the start time of a GmlTimePeriod-class.
beginPosition: Object of class "GmlTimePositionOrNULL": An object of class GmlTimePosition-class, the start time of a GmlTimePeriod-class.
calendarEraName: Object of class "character": The name of the calendar era.
duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.

duration: Object of class "character": Duration of an interval using ISO 8601 syntax for temporal length.
unit: Object of class "character": Unit parameter for a GmlTimeInterval-class.

uomLabels: Object of class "character": Unit of measurement labels as an ordered character vector for the axes in a bounding box, e.g. "deg".

upperCorner: Object of class "GmlDirectPosition": Object of class GmlDirectPosition-class, the upper (right) corner of an GmlEnvelope-class.

uom: Object of class "character": The unit of measurement in a GmlMeasure object.

value: Object of class "numeric": The actual value in a GmlMeasure object.

Methods

encodeXML signature(obj = "GmlDirectPosition", sos = "SOS") or signature(obj = "GmlEnvelope", sos = "SOS") and more: Convert the given element to an XML respresentation, and XML "encoding".

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

References


Examples

showClass("GmlDirectPosition")
showClass("GmlEnvelope")
showClass("GmlFeature")
showClass("GmlFeatureCollection")
showClass("GmlFeatureOrNULL")
showClass("GmlFeatureProperty")
showClass("GmlGeometry")
showClass("GmlLineString")
showClass("GmlPoint")
showClass("GmlPointProperty")
showClass("GmlPolygon")
showClass("GmlTimeGeometricPrimitive")
showClass("GmlTimeInstant")
showClass("GmlTimeInstantOrNULL")
showClass("GmlTimeInstantProperty")
showClass("GmlTimeInstantPropertyOrNULL")
showClass("GmlTimeInterval")
showClass("GmlTimeIntervalOrNULL")
showClass("GmlTimeObject")
showClass("GmlTimeObjectOrNULL")
showClass("GmlTimePeriod")
showClass("GmlTimePosition")
showClass("GmlTimePositionOrNULL")
showClass("GmlTimePrimitive")
# create direct position
pos1 <- GmlDirectPosition(pos = "7.0 52.0")
show(pos1)

# create envelope
env1 <- GmlEnvelope(upperCorner = pos1,
                      lowerCorner = GmlDirectPosition("6.0 51.0"))
print(env1)

# create envelope with CRS and access it
env1 <- GmlEnvelope(upperCorner = pos1,
                      lowerCorner = GmlDirectPosition("6.0 51.0"),
                      srsName = "http://www.opengis.net/def/crs/EPSG/0/4326")
sosGetCRS(env1)

# wrap elements in feature collection
GmlFeatureCollection(id = "001", featureMembers=list(pos1, env1))

# create point with ID
point1 <- GmlPoint(pos = pos1, id = "002")

# create point properties
GmlPointProperty(href = "http://link.to/point")
GmlPointProperty(point = point1)

# time interval of one day
GmlTimeInterval(interval = "1", unit = "d")

# referenced feature
GmlFeatureProperty(href = "http://link.to/feature")

# create a time position and wrap it into a time instant
timePos1 <- GmlTimePosition(time = as.POSIXct("2010-01-01"))

# create direct or referenced time instant
timeInst1 <- GmlTimeInstant(timePosition = timePos1)
timeInst1

GmlTimeInstantProperty(href = "http://link.to/timeInstant")

# create different variants of time periods
# one hour with time positions
GmlTimePeriod(beginPosition = timePos1, endPosition = GmlTimePosition(time = timePos1+3600))

# one week backwards from now
timePos <- GmlTimePosition(time = Sys.time()-(3600*24*7))
aWeekAgo <- GmlTimeInstantProperty(time = GmlTimeInstant(time = timePos))
timePos <- GmlTimePosition(time = Sys.time())
now <- GmlTimeInstantProperty(time = GmlTimeInstant(time = timePos))
GmlTimePeriod(begin = aWeekAgo, end = now)
Description

A monitoring point is the feature of interest defined for WaterML observations, i.e. a monitoring point represents the real world feature for which observations are taken. This may be, for example, the position of a stream flow sensor at a river.

Objects from the Class

Objects can be created by calls of the form `new("WmlMonitoringPoint",...)`.

Slots

- **id**: Object of class "character", the GML id of the point.
- **identifier**: Object of class "character", the identifier of the point.
- **names**: Object of class "character", the names of the point.
- **sampledFeatures**: Object of class "character", the identifiers of sampled features.
- **shape**: Object of class "SamsShape", the shape of the point.
- **timeZone**: Object of class "ANY", the time zone (if available).
- **verticalDatums**: Object of class "AN", the vertical datum (if available).

Extends

Class "GmlFeature", directly. Class "GmlFeatureOrNULL", by class "GmlFeature", distance 2. Class "GmlFeatureOrGmlFeaturePropertyOrNULL", by class "GmlFeature", distance 2.

Methods

- **sosName** signature(x = "WmlMonitoringPoint"): access the name of the point.

Examples

```
showClass("WmlMonitoringPoint")
```
Classes and Construction Functions for the OGC Namespace

Description

These classes represent elements from the OpenGIS(R) Filter Encoding Implementation Specification that are used in requests to Sensor Observation Services.

Usage

OgcBBOX(propertyName = sosDefaultSpatialOpPropertyName, envelope)
OgcContains(propertyName = sosDefaultSpatialOpPropertyName, geometry = NULL, envelope = NULL)
OgcIntersects(propertyName = sosDefaultSpatialOpPropertyName, geometry = NULL, envelope = NULL)
OgcOverlaps(propertyName = sosDefaultSpatialOpPropertyName, geometry = NULL, envelope = NULL)

Arguments

Arguments for the construction functions are as follows.

- The value for the propertyName attribute.
- geometry
- envelope

Details

These comprise spatial and temporal operations and operators which can be encoded in different ways.

The ...OrNULL classes are used to model optional slots.

Value

The value of the construction functions is an object of the respective class.

Objects from the Class

Objects can be created by calls to the respective construction functions of the form OgcBBOX( ...), OgcContains(...), or OgcIntersects.

The following classes are virtual, no objects may be created from them: OgcBinaryTemporalOp, OgcBinaryTemporalOpOrNULL, OgcComparisonOps, codeOgcComparisonOpsOrNULL, OgcSpatialOps, OgcSpatialOpsOrNULL.
Slots

propertyName: Object of class "character", the value of the propertyName attribute.
geometry: Object of class "GmlGeometry", a geometry contained in a spatial filter.
envelope: Object of class "GmlEnvelope", an envelope contained in a spatial filter.
time: Object of class "GmlTimeGeometricPrimitive", a time element contained in a temporal filter.

Extends

OgcSpatialOps: Class "OgcSpatialOpsOrNULL", directly.

Methods

encodeKVP signature(obj = "OgcBinaryTemporalOp",sos = "SOS"): Encode the given operation in key-value-pair style, see encodeKVP
encodeXML signature(obj = "OgcBBOX",sos = "SOS"): Encode the given operation in XML, see encodeXML
encodeXML signature(obj = "OgcComparisonOps",sos = "SOS"): Encode the given operation in XML, see encodeXML
encodeXML signature(obj = "OgcContains",sos = "SOS"): Encode the given operation in XML, see encodeXML
encodeXML signature(obj = "OgcIntersects",sos = "SOS"): Encode the given operation in XML, see encodeXML
encodeXML signature(obj = "OgcOverlaps",sos = "SOS"): Encode the given operation in XML, see encodeXML

Warning

The encoding functions of these classes are not completely implemented yet.

Note

This implementation of the Filter Encoding Specification is not complete.

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>
OmMeasurement

References

Schemas: http://schemas.opengis.net/filter/1.1.0/

Examples

showClass("OgcBBOX")
showClass("OgcBinarySpatialOp")
showClass("OgcBinaryTemporalOp")
showClass("OgcBinaryTemporalOpOrNULL")
showClass("OgcComparisonOps")
showClass("OgcContains")
showClass("OgcOverlaps")
showClass("OgcSpatialOps")
showClass("OgcSpatialOpsOrNULL")

OmMeasurement

Class and Construction Function for om:Measurement Elements

Description

Classes and construction functions for objects from the OGC Observations and Measurements specification.

Usage

OmMeasurement(samplingTime, procedure, observedProperty, featureOfInterest, result, metadata = NA, resultTime = NULL, resultQuality = NA, parameter = NA)

Arguments

samplingTime  See slot description.
procedure      See slot description.
observedProperty See slot description.
featureOfInterest See slot description.
result         See slot description.
metadata       See slot description.
resultTime     See slot description.
resultQuality  See slot description.
parameter      See slot description.
Details

A Measurement contains a \texttt{GmlMeasure}.

Value

The construction functions return an object of the respective class.

Objects from the Class

Objects can be created by calls to the construction function of the form \texttt{OmMeasurement(...)}.

Slots

- \texttt{featureOfInterest}: Object of class \texttt{"GmlFeature"}: A feature of any type (ISO 19109, ISO 19101), which is a representation of the observation target, being the real-world object regarding which the observation is made.
- \texttt{metadata}: Object of class \texttt{"ANY"}: Observation metadata.
- \texttt{observedProperty}: Object of class \texttt{"SwePhenomenonProperty"}: Identifies or describes the phenomenon for which the observation result provides an estimate of its value. It must be a property associated with the type of the feature of interest.
- \texttt{parameter}: Object of class \texttt{"ANY"}: A general event-specific parameter. This will typically be used to record environmental parameters, or event-specific sampling parameters that are not tightly bound to either the feature-of-interest or the procedure.
- \texttt{procedure}: Object of class \texttt{"ANY"}: The description of a process used to generate the result. It must be suitable for the observed property.
- \texttt{result}: Object of class \texttt{"ANY"}: Contains the value generated by the procedure. The type of the observation result must be consistent with the observed property, and the scale or scope for the value must be consistent with the quantity or category type.
- \texttt{resultQuality}: Object of class \texttt{"ANY"}: Event specific quality of a result.
- \texttt{resultTime}: Object of class \texttt{"GmlTimeObjectOrNULL"}: The time when the procedure associated with the observation act was applied. For some observations these are identical, in which case the resultTime may be omitted. However, there are important cases where they differ.
- \texttt{samplingTime}: Object of class \texttt{"GmlTimeObjectOrNULL"}: The time that the result applies to the feature-of-interest. This is the time usually required for geospatial analysis of the result.

Extends

\texttt{OmMeasurement}: Class \texttt{"OmObservation"}, directly. Class \texttt{"OmObservationOrNULL"}, by class \texttt{"OmObservation"}, distance 2.

Methods

\texttt{sosResult signature(obj = "OmMeasurement")}: Get the data in the measurement as a data.frame.
\texttt{sosGetCRS signature(obj = "OmMeasurement")}: Get the coordinate reference system used in the feature of interest.
Author(s)
Daniel Nuest <daniel.nuest@uni-muenster.de>

References

See Also
See also OmObservation-class, GmlMeasure.

Examples

```r
showClass("OmMeasurement")
```

---

**OmObservation-class**  
*Classes for om:Observation Elements*

**Description**
Classes and construction functions for objects from the OGC Observations and Measurements specification.

**Usage**

```r
OmObservation(samplingTime, procedure, observedProperty, featureOfInterest, result, metadata = NA, resultTime = NULL, resultQuality = NA, parameter = NA)
```

**Arguments**

- **samplingTime**  
  See slot description.

- **procedure**  
  See slot description.

- **observedProperty**  
  See slot description.

- **featureOfInterest**  
  See slot description.

- **result**  
  See slot description.

- **metadata**  
  See slot description.

- **resultTime**  
  See slot description.

- **resultQuality**  
  See slot description.

- **parameter**  
  See slot description.

- **href**  
  See slot description.

- **obs**  
  See slot description.
Details

The class `OmObservationProperty` can be used to reference to an (online) observation.
The `...OrNULL` classes are used to model optional slots.

Value

The construction functions return an object of the respective class.

Objects from the Class

Objects can be created by calls to the construction functions of the form `OmObservation(...)` and `OmObservationProperty(...)`. The following classes are virtual, no objects may be created from them: `OmObservationOrNULL-class`.

Slots

- `samplingTime`: Object of class "GmlTimeObjectOrNULL": The time that the result applies to the feature-of-interest. This is the time usually required for geospatial analysis of the result.
- `procedure`: Object of class "ANY": The description of a process used to generate the result. It must be suitable for the observed property.
- `observedProperty`: Object of class "SwePhenomenonProperty": Identifies or describes the phenomenon for which the observation result provides an estimate of its value. It must be a property associated with the type of the feature of interest.
- `featureOfInterest`: Object of class "GmlFeature": A feature of any type (ISO 19109, ISO 19101), which is a representation of the observation target, being the real-world object regarding which the observation is made.
- `result`: Object of class "ANY": Contains the value generated by the procedure. The type of the observation result must be consistent with the observed property, and the scale or scope for the value must be consistent with the quantity or category type.
- `metadata`: Object of class "ANY": Observation metadata.
- `resultTime`: Object of class "GmlTimeObjectOrNULL": The time when the procedure associated with the observation act was applied. For some observations these are identical, in which case the resultTime may be omitted. However, there are important cases where they differ.
- `resultQuality`: Object of class "ANY": Event specific quality of a result.
- `parameter`: Object of class "ANY": A general event-specific parameter. This will typically be used to record environmental parameters, or event-specific sampling parameters that are not tightly bound to either the feature-of-interest or the procedure.
- `href`: Object of class "character": Referenced observation in a `OmObservationProperty`.

Extends

`OmObservation` Class "OmObservationOrNULL", directly.

Methods

- `sosResult` signature(obj = "OmObservation"): Accessor function for the result slot of an observation or measurement.
OmObservationCollection

Author(s)
Daniel Nuest <daniel.nuest@uni-muenster.de>

References

See Also
See Also as OmMeasurement-class.

Examples
```
showClass("OmObservation")
showClass("OmObservationProperty")
showClass("OmObservationOrNULL")

OmObservationProperty(href = "http://link.to/myObservation")

# get result from an observation
## Not run:
result <- observation@result

# the accessor method also works with lists of observations
resultList <- sosResult(observation)
resultList <- sosResult(observationList)

## End(Not run)
```

OmObservationCollection

Class "OmObservationCollection"

Description
Collection of arbitrary observations.

Objects from the Class
Objects can be created by calls to the construction function of the form OmObservationCollection(...).

Slots
boundedBy: Object of class "list" containing a representation of the bounding box of the contained observations.
members: Object of class "list" containing objects of class OmObservation or OmMeasurement.
Methods

**length** signature(x = "OmObservationCollection"): Returns the number of observations in the slot members.

**sosResult** signature(obj = "OmObservationCollection"): Extract the result slots from the contained observations or measurements.

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

References


See Also

OmObservation or OmMeasurement.

Examples

showClass("OmObservationCollection")

---

OmOM_Observation-class

Class "OmOM_Observation"

Description

Classes and construction functions for objects from the OGC Observations and Measurements specification, version 2.0.

Objects from the Class

Objects can be created by calls of the form new("OmOM_Observation", ...).

Slots

phenomenonTime: Object of class "GmlTimeObjectOrNULL": The time that the result applies to the feature-of-interest. This is the time usually required for geospatial analysis of the result. Note: The phenomenonTime has been renamed to samplingTime in the O&M specification version 2.0. However, in order to be consistent with version 1.0 in SOS4R, we decided to leave the slot as is and just changed the parsing/encoding.

procedure: Object of class "ANY": The description of a process used to generate the result. It must be suitable for the observed property.
observedProperty: Object of class "SwePhenomenonProperty": Identifies or describes the phenomenon for which the observation result provides an estimate of its value. It must be a property associated with the type of the feature of interest.

featureOfInterest: Object of class "GmlFeature": A feature of any type (ISO 19109, ISO 19101), which is a representation of the observation target, being the real-world object regarding which the observation is made.

result: Object of class "ANY": Contains the value generated by the procedure. The type of the observation result must be consistent with the observed property, and the scale or scope for the value must be consistent with the quantity or category type.

metadata: Object of class "ANY": Observation metadata.

resultTime: Object of class "GmlTimeObjectOrNULL": The time when the procedure associated with the observation act was applied. For some observations these are identical, in which case the resultTime may be omitted. However, there are important cases where they differ.

resultQuality: Object of class "ANY": Event specific quality of a result.

parameter: Object of class "ANY": A general event-specific parameter. This will typically be used to record environmental parameters, or event-specific sampling parameters that are not tightly bound to either the feature-of-interest or the procedure.

Extends

Class "OmOM_ObservationOrNULL", directly.

Methods

sosCoordinates signature(obj = "OmOM_Observation"): Accessor to the spatial coordinates of an observation.

sosFeatureIds signature(obj = "OmOM_Observation"): Accessor to the feature identifiers.

sosFeaturesOfInterest signature(obj = "OmOM_Observation"): Accessor to the feature objects.

sosObservedProperties signature(obj = "OmOM_Observation"): Accessor to the observed properties in the observation.

sosProcedures signature(obj = "OmOM_Observation"): Accessor to the procedures in the observation.

sosResult signature(obj = "OmOM_Observation"): Extract the data from the observation as a data.frame.

sosTime signature(obj = "OmOM_Observation"): Accessor to the time period of the observation.

Examples

showClass("OmOM_Observation")
Classes and Construction Functions for Elements of the OWS Namesp ace

Description

These classes represent elements from the OGC Web Services Common Specification and the OGC Web Services Common Standard.

Usage

OwsCapabilities(version, updateSequence = NA, owsVersion = sosDefaultGetCapOwsVersion, identification = NULL, provider = NULL, operations = NULL, contents = NULL, languages = NULL)
OwsException(exceptionCode, exceptionText = c(), locator = as.character(NA))
OwsExceptionReport(version, lang = as.character(NA), exceptions = list(NA))
OwsGetCapabilities(service, acceptVersions, sections = sosDefaultGetCapSections, acceptFormats = sosDefaultGetCapAcceptFormats, updateSequence = c(as.character(NA)), owsVersion = sosDefaultGetCapOwsVersion, acceptLanguages = c(NA))
OwsOperation(name, DCPs, parameters = list(NA), constraints = list(NA), metadata = list(NA))
OwsOperationsMetadata(operations, parameters = list(NA), constraints = list(NA), extendedCapabilities = xml2::xml_missing())
OwsRange(minimumValue = as.character(NA), maximumValue = as.character(NA), rangeClosure = as.character(NA), spacing = as.character(NA))
OwsServiceProvider(providerName, providerSite = as.character(NA), serviceContact = xml2::xml_missing())
OwsServiceIdentification(serviceType, serviceTypeVersion, profile = c(NA), title, abstract = c(NA), keywords = c(NA), fees = as.character(NA), accessConstraints = c(NA))

Arguments

abstract Brief narrative description of this server, normally available for display to a human.
acceptFormats Unordered character vector of zero or more response formats desired by client, with preferred formats listed first.
acceptLanguages Unordered character vector of zero or more languages desired by client, with preferred formats listed first. Only OWS 2.0.0!
acceptVersions Comma-separated prioritized sequence of one or more specification versions accepted by client, with preferred versions listed first.
accessConstraints Access constraints that should be observed to assure the protection of privacy or intellectual property, and any other restrictions on retrieving or using data from or otherwise using a server.
constraints  Constraint on valid domain of a non-parameter quantity that applies to an operation which a server implements.

contents  The provider section of a capabilities document, object of class OwsContentsOrNULL.

DCPs  Information for a Distributed Computing Platform (DCP) supported for an operation.

exceptionCode  The code attribute of an OWS Exception, see OwsExceptionsData.

exceptions  The list of OwsException in a OwsExceptionReport.

exceptionText  The text element of an OWS Exception, see OwsExceptionsData.

extendedCapabilities  The possible contents of the ExtendedCapabilities subsection are not specified by the SOS standard.

fees  Fees and terms for using a server, including the monetary units as specified in ISO 4217.

identification  The identification section of a capabilities document, object of class OwsServiceIdentificationOrNULL.

keywords  Unordered list of one or more commonly used or formalised word(s) or phrase(s) used to describe a server.

lang  The code attribute of an OWS Exception.

languages  The languages section of a capabilities document, currently an object of class XMLAbstractNode.

locator  The locator attribute of an OWS Exception, see OwsExceptionsData.

maximumValue  Maximum value of a range (numeric parameter).

metadata  Metadata about an operation and its implementation.

minimumValue  Minimum value of a range (numeric parameter).

name  Name of an operation (request) (for example, GetCapabilities).

operations  A list of objects of class OwsOperation in a OperationsMetadata object. The provider section of a capabilities document.

owsVersion  The used OWS specification version.

parameters  Parameter valid domain that applies to an operation which a server implements.

profile  Identifier of OGC Web Service (OWS) Application Profile.

providerName  Unique identifier for service provider organization.

providerSite  Reference to the most relevant web site of a service provider.

provider  The provider section of a capabilities document, object of class OwsServiceProviderOrNULL.

rangeClosure  Specifies which of minimum and maximum values are included in this range; include when not default of “closed” range. Possible values are closed, open, open-closed, or closed-open.

sections  Unordered character vector of zero or more names of sections of service metadata document to be returned in service metadata document.

serviceContact  Information for contacting service provider.

service  Service type identifier text.

serviceType  A service type name from registry of services.
serviceTypeVersion
  Version of a service type implemented by a server.

spacing
  Regular distance or spacing between allowed values in this range; include when
  range is not continuous.

title
  Title of a server, normally used for display to a human.

updateSequence
  Service metadata document version, value is "increased" whenever any change
  is made in complete service metadata document. This can be used to request a
  certain version of a metadata document. Parameter is found in both request and
  reponse, but may not be supported by a service.

version
  The version of the document.

Details

OwsServiceOperation is the top class which is eventually put into the request method, sosRequest(...).

Classes ending in ...OrNULL are used to model optional slots.

Objects from the Class

Objects can be created by calling the construction functions, e.g. in the form OwsCapabilities_1.1.0(...),
OwsContents(...) or OwsException(...).

The following classes are virtual and no objects may be created from it: OwsContentsOrNULL,
OwsServiceIdentificationOrNULL, OwsServiceProviderOrNULL, OwsOperationsMetadataOrNULL.

Slots

Capabilities:

contents: Object of class "OwsContentsOrNULL", the contents section of a capabilities document.

identification: Object of class "OwsServiceIdentificationOrNULL", the identification section
  of a capabilities document.

languages: Object of class "XMLAbstractNode", the languages section of a capabilities document,
  only OWS 2.0.0!

operations: Object of class "OwsOperationsMetadataOrNULL", the operations section of capa-
  bilities document.

owsVersion: Object of class "character", the used version of OWS.

provider: Object of class "OwsServiceProviderOrNULL", the provider section of a capabilities
document.

updateSequence: Object of class "character", the updateSequence attribute of a capabilities
document.

version: Object of class "character", the version of the described service in a capabilities doc-
  ument.

dcode: Object of class "character", the code attribute of an OWS Exception, see OwsExceptionsData.

text: Object of class "vector", the text of an OWS Exception, see OwsExceptionsData.

locator: Object of class "character", the locator attribute of an OWS Exception, see OwsExceptionsData.
version: Object of class "character", the version of an OWS Exception, see OwsExceptionsData.
lang: Object of class "character", the code attribute of an OWS Exception, see OwsExceptionsData.
exceptions: Object of class "list", the list of OwsException in a OwsExceptionReport.
sections: Object of class "vector", unordered character vector of zero or more names of sections of service metadata document to be returned in service metadata document.
acceptFormats: Object of class "vector", unordered character vector of zero or more response formats desired by client, with preferred formats listed first.
updateSequence: Object of class "vector", service metadata document version.
acceptVersions: Object of class "character", comma-separated prioritized sequence of one or more specification versions accepted by client, with preferred versions listed first.
service: Object of class "character", the name of the service.
request: Object of class "character", the name of the operation/request.
acceptLanguages: Object of class "vector", an unordered character vector of zero or more languages desired by client, with preferred formats listed first. Only OWS 2.0.0!
name: Object of class "character", name of an operation (request) (for example, GetCapabilities).
DCPs: Object of class "list", information for a Distributed Computing Platform (DCP) supported for an operation.
parameters: Object of class "list", parameter valid domain that applies to an operation which a server implements.
constraints: Object of class "list", Constraint on valid domain of a non-parameter quantity that applies to an operation which a server implements.
metadata: Object of class "list", metadata about an operation and its implementation.
operations: Object of class "list", a list of objects of class OwsOperation in a OperationsMetadata object.
extendedCapabilities: Object of class "XMLAbstractNode", the possible contents of the ExtendedCapabilities subsection.
minimumValue: Object of class "character", minimum value of a range (numeric parameter).
maximumValue: Object of class "character", maximum value of a range (numeric parameter).
rangeClosure: Object of class "character", specifies which of minimum and maximum values are included in this range.
spacing: Object of class "character", regular distance or spacing between allowed values in this range; included when range is not continuous.
serviceType: Object of class "character", the service type name from registry of services.
serviceTypeVersion: Object of class "vector", version of a service type implemented by the server.
profile: Object of class "vector", identifier of OGC Web Service (OWS) Application Profile.
title: Object of class "vector", title of the server, normally used for display to a human.
abstract: Object of class "vector", brief narrative description of this server, normally available for display to a human.
keywords: Object of class "vector", unordered list of one or more commonly used or formalised word(s) or phrase(s) used to describe a server.
fees: Object of class "character", fees and terms for using a server, including the monetary units as specified in ISO 4217.

accessConstraints: Object of class "vector", access constraints that should be observed to assure the protection of privacy or intellectual property, and any other restrictions on retrieving or using data from or otherwise using a server.

providerName: Object of class "character", unique identifier for service provider organization.

providerSite: Object of class "character", reference to the most relevant web site of a service provider.

serviceContact: Object of class "XMLAbstractNode", information for contacting service provider.

**Extends**

- **OwsCapabilities_1.1.0**: Class "OwsCapabilities", directly.
- **OwsCapabilities_2.0.0**: Class "OwsCapabilities_1.1.0", directly. Class "OwsCapabilities", by class "OwsCapabilities_1.1.0", distance 2.
- **OwsGetCapabilities_1.1.0**: Class "OwsGetCapabilities", directly. Class "OwsServiceOperation", by class "OwsGetCapabilities", distance 2.
- **OwsGetCapabilities_2.0.0**: Class "OwsGetCapabilities_1.1.0", directly. Class "OwsGetCapabilities", by class "OwsGetCapabilities_1.1.0", distance 2. Class "OwsServiceOperation", by class "OwsGetCapabilities_1.1.0", distance 3.
- **OwsServiceIdentification**: Class "OwsServiceIdentificationOrNULL", directly.
- **OwsServiceProvider**: Class "OwsServiceProviderOrNULL", directly.

**Methods**

- **show** signature(object = "<NAME OF CLASS>"): Shows a human readable version of the object.
- **encodeRequestKVP** signature(obj = "OwsGetCapabilities"): See link{encodeRequestKVP}.
- **encodeRequestSOAP** signature(obj = "OwsGetCapabilities"): See link{encodeRequestSOAP}.
- **encodeRequestXML** signature(obj = "OwsGetCapabilities"): See link{encodeRequestXML}.
- **encodeRequestKVP** signature(obj = "OwsGetCapabilities_2.0.0"): See link{encodeRequestKVP}.
- **encodeRequestKVP** signature(obj = "OwsGetCapabilities_1.1.0"): See link{encodeRequestKVP}.
- **sosRequest** signature(sos = "SOS", request = "OwsServiceOperation", verbose = "logical", inspect = "logical"): Send the given operation as a request to the given SOS.

**Author(s)**

Daniel Nuest <daniel.nuest@uni-muenster.de>

**References**

Whiteside A. (Ed.), OGC Web Services Common Specification, Open Geospatial Consortium Inc., OGC 06-121r3, Version: 1.1.0 with Corrigendum 1

Whiteside A., Greenwood, J. (Eds.), OGC Web Services Common Standard, Open Geospatial Consortium Inc., OGC 06-121r9, Version: 2.0.0
Examples

```
showClass("OwsCapabilities_1.1.0")
showClass("OwsCapabilities_2.0.0")
showClass("OwsCapabilities")
showClass("OwsContents")
showClass("OwsContentsOrNULL")
showClass("OwsException")
showClass("OwsExceptionReport")
showClass("OwsGetCapabilities_1.1.0")
showClass("OwsGetCapabilities_2.0.0")
showClass("OwsGetCapabilities")
showClass("OwsOperation")
showClass("OwsOperationsMetadata")
showClass("OwsRange")
showClass("OwsServiceIdentification")
showClass("OwsServiceIdentificationOrNULL")
showClass("OwsServiceOperation")
showClass("OwsServiceProvider")
showClass("OwsServiceProviderOrNULL")
```

---

**parse**

*Parsing Functions for XML Documents and Elements*

---

**Description**

The functions decode a given XML object to an R representation, which can be an object of a specific class, a list, a named character vector, ...

**Usage**

```
parseCSV(obj, verbose = FALSE)
parseFile(sos, file, verbose = FALSE, ...)
parsesCategoryObservation(obj, sos, verbose = FALSE)
parseComplexObservation(obj, sos, verbose = FALSE)
parseComponent(obj, sos, verbose = FALSE)
parseCompositePhenomenon(obj, sos, verbose = FALSE)
parseCountObservation(obj, sos, verbose = FALSE)
parseDataArray(obj, sos, verbose = FALSE)
parseElementType(obj, sos, verbose = FALSE)
parseEncoding(obj, sos, verbose = FALSE)
parseEnvelope(obj, sos, namespaces = xml2::xml_ns(obj), verbose = FALSE)
parseFeatureCollection(obj, sos)
puseStateField(obj, sos, verbose = FALSE)
purateOII(obj, sos, verbose = FALSE)
purateGeometryObservation(obj, sos, verbose = FALSE)
purateMeasurement(obj)
purateNoParsing(obj, sos)
```
parseObservation(obj, sos, verbose = FALSE)
parseObservationCollection(obj, sos, verbose)
parseOM(obj, sos, verbose = FALSE)
parseOwsException(obj, sos)
parseOwsExceptionReport(obj, sos, verbose = FALSE)
parseOwsOperation(obj, sos)
parseOwsRange(obj)
parseOwsServiceIdentification(obj, sos)
parseOwsServiceProvider(obj, sos)
parsePhenomenonProperty(obj, sos, verbose = FALSE)
parsePoint(obj, sos)
parsesPosition(obj, sos)
parsedResult(obj, sos, verbose = FALSE)
parsingSamplingPoint(obj, sos)
parsingSensorML(obj, sos, verbose = FALSE)
parsesSosCapabilities(obj, sos)
parsesSosFilterCapabilities(obj, sos)
parsesSosObservationOffering(obj, sos)
parseTemporalObservation(obj, sos, verbose = FALSE)
parsedTextBlock(obj)
parsesTimeGeometricPrimitiveFromParent(obj, sos)
parsesTimeInstant(obj, sos)
parsesTimeInstantProperty(obj, sos)
parsesTimePeriod(obj, sos)
parsesTimePosition(obj, sos)
parsesTimeObject(obj, sos, verbose = FALSE)
parsesTruthObservation(obj, sos, verbose = FALSE)
parsesValues(values, fields, encoding, sos, verbose = FALSE)

sosIsListFieldAvailable(listField)

gmlGetReferencedNode(sos, doc, node, verbose = FALSE)
gmlIsNodeReferenced(sos, node)

Arguments

obj
sos
verbose
namespaces
values
fields
file
encoding

The object to decode, normally objects of either xml_document.
An object of class SOS-class, which may be utilized/required by some parsing
functions to access other parsing functions or encoding information.
A boolean value indication whether information is printed out to the console
during the process - potentially a lot of output!
A vector of namespace prefixes and definitions to use.
The values to be parsed in parseValues(...).
Field information in parseValues(...), a named list.
Name of the file to be parsed in sosParse(...).
Encoding information in parseValues(...), an object of class SweTextBlock.
Additional arguments that are parsed to `xml2::read_xml()`.

**listField** Object checked to be of type list and at least one item long.

**doc** XML document.

**node** XML node.

**Details**

The naming of the functions follow the following rule: `parse[optional: namespace prefix][name of the XML element to be parsed]`

Not all parsing function that have a SOS object or verbose in their signature, but few actually use it at this points of development. Some of the parsing functions are **exchangeable** when creating a new SOS connection. Please see the examples!

Functions parsing time rely on the slot `timeFormat` of SOS objects.

`parseOM` is a special function in the respect that it matches sub parsing function depending on an objects xmlName from the list of the given SOS's parsing functions.

`parseNoParsing` is a convenience function that directly returns the object without any changes.

`sosParse` allows parsing of files for all elements that have a parsers registered with the given SOS.

The parsing uses an internal cache to store within-document references to other nodes.

**Value**

An objects of a specific class depending on the parsing method and the passed object, possibly even lists or named character vectors.

**Warning**

Functions might result in error if parsed an object of the wrong type, because that is normally not checked.

Some of the functions are placeholders for future implementations!

**Author(s)**

Daniel Nuest <daniel.nuest@uni-muenster.de>

**See Also**

*SosParsingFunctions*, *sosParsers-methods*

**Examples**

```r
# parsing a XML string to an exception report object
er.doc <- xml2::read_xml(paste0("<ows:ExceptionReport xmlns:ows="http://www.opengis.net/ows/1.1\"",
  " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.0.0\"",
  " xsi:schemaLocation="http://schemas.opengis.net/ows/1.1.0/owsExceptionReport.xsd"",
  "<ows:Exception exceptionCode="VersionNegotiationFailed" locator="AcceptVersions\"">",
  "<ows:ExceptionText>The parameter 'AcceptVersions' does not contain the version",
  " of this SOS: '1.0.0'</ows:ExceptionText></ows:Exception></ows:ExceptionReport>"))
```
er.parsed <- parseOwsExceptionReport(er.doc, sos = SOS_Test())
print(er.parsed)
utils::str(er.parsed)

## Not run:
# save and re-parse an observation from file
obsId <- getObservationById(sos = mySOS, observationId = "o_3508493",
saveOriginal = TRUE)
.files <- list.files(getwd())
.startWithO_ <- .files %in% grep("o_", .files, value=TRUE)
.observationFiles <- subset(.files, .startWithO_)

obsId <- parseFile(sos = mySOS, file = .observationFiles[[1]])

# list caches
ls(get("sos4R_caches"))

## End(Not run)

---

**phenomena**

*Function retrieve phenomena of a SOS v2.0*

**Description**

This function queries a SOS v2.0 for all its phenomena

**Usage**

```r
phenomena(sos, includeTemporalBBox = FALSE, includeSiteId = FALSE, ...)
```

**Arguments**

- `sos` An SOS object from a call to `SOS` containing the URL of a SOS.
- `includeTemporalBBox` Whether the observation containing timespan of each phenomenon shall also be reported.
- `includeSiteId` Whether the sites shall be reported at which each phenomenon has been observed.
- `...` Named arguments forwarded to other wrapper functions internally.

**Value**

If only the argument `sos` is provided, this function returns a character vector with the phenomenon ids.

If `timeInfo = TRUE`, the result is a data.frame where the character vector of phenomena is combined with a character columns `beginTime` and `endTime`. Note that these two columns indicate the
very beginning and end of any time series recorded for each phenomenon across all sites and does not imply any continuity.

If sites = TRUE, the result is a data.frame where the character vector of phenomena is combined with a character column siteID. Phenomena that have been observed at several sites will be replicated for each site. If also timeInfo = TRUE, the data.frame will have character columns beginTime and endTime per phenomenon and site. Note that these two columns indicate the very beginning and end of any time series recorded for each phenomenon at each site, but does still not imply to be continuous series.

Author(s)
Benedikt Graeler, Eike Hinderk Juerrens

Examples

mySos <- SOS(url = "https://climate-sos.niwa.co.nz/",
             binding = "KVP", useDCPs = FALSE, version = "2.0.0")

phenomena(sos = mySos)

Description
Classes and construction functions for elements from the OGC specification “Observations and Measurements - Part 2 - Sampling Features”.

Usage

SaSamplingPoint(sampledFeatures, position, relatedObservation = list(NA),
               relatedSamplingFeature = list(NA), surveyDetails = NA, id = NA_character_)

Arguments

sampledFeatures
position
relatedObservation
relatedSamplingFeature
surveyDetails
id

Value
Construction functions: An object of the respective class.
Objects from the Class

Objects can be created by calls to the construction functions of the form `SaSamplingPoint(...)`.

Slots

- **sampledFeatures**: Object of class "list" which contains the sampled features.
- **position**: Object of class "GmlPointProperty" which contains the position of a feature.
- **relatedObservation**: Object of class "list" which contains identifiers of related observations.
- **relatedSamplingFeature**: Object of class "list" which contains identifiers of related sampling features.
- **surveyDetails**: Object of class "ANY" which can contain information about survey details (which are currently no modeled in an R class).
- **id**: The identifier of a sampling elements (object of class "character").
- **shape**: Object of class "ANY" which could be used to model the shape of a sampling surface.
- **area**: Object of class "ANY" which could be used to model the area of a sampling surface.

Extends

`SaSamplingPoint` and `SaSamplingSurface`: Class "GmlFeature", directly. Class "GmlFeatureOrNULL", by class "GmlFeature", distance 2.

Methods

- `show` signature(`object = "SaSamplingPoint"`): ...
- `show` signature(`object = "SaSamplingSurface"`): ...
- `sosCoordinates` signature(`object = "SaSamplingPoint"`): access to the coordinates of the feature.
- `sosFeatureIds` signature(`object = "SaSamplingPoint"`): access to the identifier of the feature.
- `sosFeaturesOfInterest` signature(`object = "SaSamplingPoint"`): access to the feature of interest, i.e. the point itself.

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

References

Simon Cox (Ed.), Observations and Measurements - Part 2 - Sampling Features, OGC 07-002r3

Examples

```r
showClass("SaSamplingPoint")

# create sampling point
SaSamplingPoint(sampledFeatures = list("feature1", "feature2"),
                position = GmlPointProperty(href = "http://link.to/point"))
```
Classes and creation functions for Sampling Features

Description

Sampling Feature classes.

SamsShape represents the geometry of a spatial sampling feature, that can be used as a feature of interest. Currently, only points are supported.

Objects from the Class

Objects can be created by calls to the creation functions:

SamsSamplingFeature(...)
SamsShape(...)

Slots

identifier: Object of class "character" ~
name: Object of class "character" ~
type: Object of class "character" ~
sampledFeature: Object of class "character" ~
shape: Object of class "SamsShape" ~
id: Object of class "character" ~
point: Object of class "GmlPoint" ~
id: Object of class "character" ~
srsName: Object of class "character" ~
srsDimension: Object of class "integer" ~
axisLabels: Object of class "character" ~
uomLabels: Object of class "character" ~

Extends

SamsShape extends "GmlGeometry", directly.
Methods

**sosCoordinates** signature(obj = "SamsSamplingFeature"): access to the coordinates of the feature.

**sosFeatureIds** signature(obj = "SamsSamplingFeature"): access to the identifier of the feature.

**sosFeaturesOfInterest** signature(obj = "SamsSamplingFeature"): access the feature itself.

**sosName** signature(obj = "SamsSamplingFeature"): access a human-readable name of the feature.

Note

Schema: http://schemas.opengis.net/sampling/2.0/samplingFeature.xsd

Author(s)

Daniel Nuest

References

https://www.opengeospatial.org/standards/om

Examples

```
s showError("SamsSamplingFeature")
s showError("SamsShape")
```

---

**siteList**

This function queries a SOS v2.0 for its sites and returns them as data.frame

Description

Information about the registered sites in a SOS can be queried using this function. This function allows for different levels of detail and returns a vector or data.frame.

Usage

```
siteList(sos,
        empty = FALSE,
        begin = NA,
        end = NA,
        phenomena = list(),
        includePhenomena = FALSE,
        includeTemporalBBox = FALSE,
        ...)
```
Arguments

sos
A SOS object from a call to `SOS` containing the URL of a SOS.

empty
Whether also empty sites shall be included in the response.

begin
Object of class POSIXt. Note that creating timestamps with `as.POSIXct(..)` will use your local timezone.

dead
Object of class POSIXt. Note that creating timestamps with `as.POSIXct(..)` will use your local timezone.

phenomena
A list, vector or one-column data.frame with characters identifying the relevant phenomena.

includePhenomena
Whether the phenomena shall also be reported in the result.

includeTemporalBBox
Whether the observation containing timespan of each phenomenon shall also be reported.

... Named parameters passed on to other wrapper functions.

Value

If only the argument `sos` is provided and the flag `empty` is set to `TRUE` or `FALSE`, this function returns a character vector with the site ids.

If `listPhenomena = TRUE`, the data.frame contains the column containing the site ids and a character column phenomenon where sites with several phenomena are repeated for each of their phenomenon.

If `includeTemporalBBox = TRUE`, the data.frame also contains two character columns `beginTime` and `endTime` indicating per site id and phenomenon its earliest and latest observation time point.

Author(s)

Benedikt Graeler, Eike Hinderk Juerrens

Examples

```r
## Not run:
mySos <- SOS(url = "https://climate-sos.niwa.co.nz/",
             binding = "KVP", useDCPs = FALSE, version = "2.0.0")
siteList(sos = mySos)

## End(Not run)
```
This function queries a SOS v2.0 for its sites and returns them as SpatialPointsDataFrame

Description

Information about the registered sites in a SOS can be queried using this function. This function allows for different levels of detail and returns a SpatialPointsDataFrame.

Usage

sites(sos, begin = NA, end = NA, phenomena = list(), empty = FALSE, includePhenomena = FALSE, includeTemporalBBox = FALSE, ...)

Arguments

sos A SOS object from a call to SOS containing the URL of a SOS.
begin Object of class POSIXt. Note that creating timestamps with as.POSIXct(..) will use your local timezone.
end Object of class POSIXt. Note that creating timestamps with as.POSIXct(..) will use your local timezone.
phenomena A list, vector or one-column data.frame with characters identifying the relevant phenomena.
empty Whether also empty sites shall be included in the response.
includePhenomena Whether the phenomena shall also be reported in the result.
includeTemporalBBox Whether the observation containing timespan of each phenomenon shall also be reported.
... Named parameters passed on to other wrapper functions.

Value

A SpatialPointsDataFrame containing all sites that match the filter conditions with corresponding metadata defined via the arguments. If empty = TRUE, the data slot contains a logical column 'empty' indicating whether any data has been observed for this site. If listPhenomena = TRUE, the data.frame contains a logical column per phenomenon indicating whether this specific phenomenon has been observed at that site. If includeTemporalBBox = TRUE, the data.frame contains a two column data.frame with beginTime and endTime per phenomenon.
Author(s)
Benedikt Graeler, Eike Hinderk Juerrens

Examples
mySos <- SOS(url = "https://climate-sos.niwa.co.nz/",
    binding = "KVP", useDCPs = FALSE, version = "2.0.0")
sites(sos = mySos)

Classes of the Namespace sml

Description
Classes, construction functions, and accessor functions for elements from the OGC specification “OpenGIS(R) Sensor Model Language (SensorML) Implementation Specification”. The only class at the moment is "SensorML" which wraps an "XMLInternalDocument" and some additional information. This strongly depends on the SensorML Profile for Discovery to find the respective parameters.

Objects from the Class
Objects can be created by calls to the construction method in the form SensorML(...).

Slots
boundedBy: Object of class "matrix", the bounding box of the sensor.
coords: Object of class "data.frame", holds the position of the sensor.
description: Object of class "character", a normal text description of the sensor.
id: Object of class "character", the main identifier of the sensor.
name: Object of class "character", a naming identifier of the sensor.
validTime: Object of class "GmlTimeObjectOrNULL", the time period or instance for which the description is valid (only SOS 2.0.0).
xml: Object of class "XMLInternalDocument", holds the XML representation of the sensor description.

Methods
show signature(object = "SensorML"): Prints a short statement to the command line.
plot signature(object = "SensorML"): Plots the sensor using coercion to an object of class "Spatial".

Author(s)
Daniel Nuest <daniel.nuest@uni-muenster.de>
SML

References


See Also
DescribeSensor

Examples

showClass("SensorML")

mySOS <- SOS(url = "http://sensorweb.demo.52north.org/sensorwebtestbed/service/kvp", binding = "KVP")
mySensor <- describeSensor(sos = mySOS, procedure = sosProcedures(mySOS)[[1]], outputFormat = 'text/xml; subtype="sensorML/1.0.1"', # space is needed!
class(mySensor)
print(mySensor)

sosId(mySensor)
sosName(mySensor)
sosBoundedBy(mySensor)
sosCoordinates(mySensor)
sosGetCRS(mySensor)

# valid time from SOS 2.0.0
mySOS2 <- SOS(url = "http://sensorweb.demo.52north.org/sensorwebtestbed/service/kvp", binding = "KVP", version = "2.0.0")
mySensor2 <- describeSensor(sos = mySOS2, procedure = sosProcedures(mySOS2)[[1]], outputFormat = 'text/xml; subtype="sensorML/1.0.1"')
sosTime(mySensor2)

## Not run:
plot(mySensor)

## End(Not run)
Class, and Construction and Accessor Functions for "SOS"

Description

Base class of a connection to a Sensor Observation Service.

Usage

SOS(url,
   binding = SosDefaultBinding(),
   version = sos100_version,
   parsers = SosParsingFunctions(),
   encoders = SosEncodingFunctions(),
   dataFieldConverters = SosDataFieldConvertingFunctions(),
   timeFormat = sosDefaultTimeFormat, verboseOutput = FALSE,
   switchCoordinates = FALSE,
   useDCPs = TRUE,
   dcpFilter = SosDefaultDCPs(),
   additionalKVPs = list(),
   namespaces = SosAllNamespaces(version = version),
   ...)

SOS_Test(name = "test",
   binding = SosDefaultBinding(),
   version = sos100_version,
   parsers = SosParsingFunctions(),
   encoders = SosEncodingFunctions(),
   dataFieldConverters = SosDataFieldConvertingFunctions(),
   timeFormat = sosDefaultTimeFormat,
   verboseOutput = FALSE,
   switchCoordinates = FALSE,
   useDCPs = TRUE,
   dcpFilter = SosDefaultDCPs(),
   additionalKVPs = list(),
   namespaces = SosAllNamespaces(version = version),
   ...)

Arguments

url See the corresponding slot description.
binding See the corresponding slot description.
version See the corresponding slot description.
parsers See the corresponding slot description.
encoders See the corresponding slot description.
Details

From the introduction of the specification document: “The goal of SOS is to provide access to observations from sensors and sensor systems in a standard way that is consistent for all sensor systems including remote, in-situ, fixed and mobile sensors.”

Value

The construction functions returns an object of class `SOS-class`.

Objects from the Class

Objects can be created by calls to the construction function of the form `SOS(...)`. Object from the class can be used in calls to function for metadata retrieval of sensors (link{describeSensor-methods}) and observation data queries (link{getObservation-methods} and link{getObservationById-methods})

Slots

url: Object of class "character": The endpoint of the service, e.g. `http://myUrl.org/SOS1/sos`.

binding: Object of class "character": The binding, or transport protocol, see `SosSupportedBindings` for available ones.

version: Object of class "character": The connected service's version, e.g. "1.0.0".

capabilities: Object of class "OwsCapabilities" ~~

parsers: Object of class "list": A list of named functions for parsing of SOS responses.

encoders: Object of class "list": A list of named functions for encoding of SOS requests.

dataFieldConverters: Object of class "list": A list of named functions to be used by the parsing methods to convert data values to the correct R type, see `SosDataFieldConvertingFunctions` for the default functions and how to add your own converters.

timeFormat: Object of class "character": The time format to be used or decoding and encoding time character strings to and from POSIXt classes.
verboseOutput: Object of class "logical": Trigger parameter for extensive debugging information on the console.

switchCoordinates: Object of class "logical": Trigger enabling switching of lat/long during parsing stage.

useDCPs: Object of class "logical": Trigger for not using DCP endpoints from the capabilities but just the URL defined when creating the SOS connection. If there is a specific endpoint for a specific binding the user must make sure the configured binding and DCP match.

dcpFilter: Object of class "list": Named list of regular expressions to be applied (using grep) to DCPs if there is more than one for the chosen binding.

additionalKVPs: A nam of class "list": Named list of key-value-pairs to be appended to an KVP request.

Methods

sosAbstract signature(obj = "SOS"): ...

sosBinding signature(sos = "SOS"): The protocol used for a connection to the service.

sosCapabilitiesDocumentOriginal signature(sos = "SOS"): To retrieve the full original service metadata document.

sosCaps signature(sos = "SOS"): ...

sosContents signature(sos = "SOS"): ...

sosDataFieldConverters signature(sos = "SOS"): ...

sosTime signature(obj = "SOS"): ...

sosTitle signature(obj = "SOS"): ...

sosOperations signature(sos = "..."): ...

sosGetCRS signature(obj = "character"): ...

sosGetDCP signature(sos = "SOS", operation = "character", type = "character"): Get the distributed computing platform URL for the given operation and method type. If type is missing, the function returns all available DCPs.

sosEncoders signature(sos = "SOS"): ...

sosFeatureIds signature(obj = "list"): ...

sosFeaturesOfInterest signature(obj = "list"): ...

sosGetCRS signature(obj = "character"): Get an object of class sp:::CRS for a given OGC URN depicting a reference system, like urn:ogc:def:crs:EPSG:1000.

sosName signature(obj = "..."): ...

sosObservedProperties signature(sos = "SOS"): ...

sosOffering signature(sos = "SOS", offeringId = "character"): ...

sosOfferingIds signature(sos = "SOS"): ...

sosOfferings signature(sos = "SOS"): ...

sosOperation signature(sos = "SOS", operationName = "character"): ...

sosOperationsMetadata signature(sos = "SOS"): ...
sosParsers signature(sos = "SOS"): ...

sosProcedures signature(sos = "SOS"): Accessor function for the procedures of a SOS (via list in capabilities of GetObservation operation) or a SosObservationOffering.

sosResponseFormats signature(sos = "SOS"): TBD: add missing signatures ...

sosResponseMode signature(sos = "SOS"): ...

sosResultModels signature(sos = "SOS"): ...

sosServiceIdentification signature(sos = "SOS"): ...

sosServiceProvider signature(sos = "SOS"): ...

sosSrsName signature(sos = "SOS"): ...

sosSwitchCoordinates signature(sos = "SOS"): ...

sosTimeFormat signature(sos = "SOS"): ...

sosUrl signature(sos = "SOS"): ...

sosVersion signature(sos = "SOS"): ...

sosTime signature(obj = "SOS"): Accessor function for the event time period from the GetObservation operations metadata.

sosFilter_Capabilities signature(sos = "SOS"): Accessor function for the filter capabilities of a SOS object.

describeSensor signature(sos = "SOS_2.0.0", procedure = "character"): ...

getCapabilities signature(sos = "SOS_2.0.0"): ...

getDataAvailability signature(sos = "SOS_2.0.0"): ...

getFeatureOfInterest signature(sos = "SOS_2.0.0", featureOfInterest = "ANY"): ...

getFeatureOfInterest signature(sos = "SOS_2.0.0", featureOfInterest = "character"): ...

getObservationById signature(sos = "SOS_2.0.0", observationId = "character"): ...

getObservation signature(sos = "SOS_2.0.0", offering = "character"): ...

getObservation signature(sos = "SOS_2.0.0", offering = "SosObservationOffering_2.0.0"): ...

sosBinding signature(sos = "SOS_2.0.0"): ...

sosRequest signature(sos = "SOS_2.0.0", request = "OwsServiceOperation", verbose = "logical", inspect = "logical"): ...

sosUrl signature(sos = "SOS_2.0.0"): access the endpoint of the SOS object.

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

References

Na, A., Priest, M. (Eds.), Sensor Observation Service, Open Geospatial Consortium Inc., OGC 06-009r6, Version: 1.0

The document is available for download at https://www.opengeospatial.org/standards/sos.
See Also

See also creation function `SOS` and the package vignette for general description of use.

Examples

```r
showClass("SOS")
showClass("SOS_2.0.0")

# create a SOS connection
mySOS <- SOS(url = "http://sensorweb.demo.52north.org/sensorwebtestbed/service/kvp",
             binding = "KVP")

# create the URL to a GET request for GetCapabilities
sosCapabilitiesUrl(mySOS)

# access details of the SOS connection and it's metadata
sosContents(mySOS)
sosTime(mySOS)
sosFeaturesOfInterest(mySOS)
sosBinding(mySOS)

## Not run:
# create a SOS connection with a specific connection method and time format
mysos <- SOS(url = "http://mysos.org/sos",
             binding = "KVP", timeFormat = "%Y-%m-%d")

# turn on verbose output for all methods and functions
SOS(url = "http://mysos.org/sos", verboseOutput = TRUE)

# get the meaning of an exception code
sosExceptionCodeMeaning(ex@exceptionCode)

# create a CRS object from a URN CRS string
sosGetCRS("urn:ogc:def:crs:EPSG:4217")

# create a SOS for a specific binding at a specific endpoint
SOS(url = "http://localhost:8080/52n-sos/sos/pox", binding = "POX",
    useDCPs = FALSE)

# create a SOS using only the DCPs from the capabilities that match a specific
# pattern with the default binding
SOS(url = "http://localhost:8080/52n-sos/sos/service",
    dcpFilter = list("POX" = "/pox"))

## End(Not run)
```
Description

The SOS comes with three possible methods of transferring data, HTTP GET, HTTP POST and SOAP.

Details

The **POST** binding is described in the official SOS specification and should be the default method.


It contains some special encoding for bounding boxes, as the only spatial filter, and time periods, as the only temporal filter.

The **SOAP** binding is not official with regards to the spec, and also not implemented yet.

The connection method can be changed on creation of a SOS object.

References


See Also

[SosSupportedBindings](#)

Examples

```r
# HTTP connection methods supported by this sos4R implementation
supported <- SosSupportedBindings()
supported

## Not run:
sos <- SOS("http://sosurl.org/", binding = "KVP")
## End(Not run)
```
Arguments

version: The version of the service.
updateSequence: Service metadata document version, value is "increased" whenever any change is made in complete service metadata document. This can be used to request a certain version of a metadata document. Parameter is found in both request and response, but may not be supported by a service.
owsVersion: The used OWS specification version.
identification: The identification section of a capabilities document, object of class OwsServiceIdentification.
provider: The provider section of a capabilities document, object of class OwsServiceProvider.
operations: A list of objects of class OwsOperation in an OperationsMetadata object. The provider section of a capabilities document.
filterCapabilities: An object of class SosFilter_Capabilities containing the filter capabilities of a service.
contents: The provider section of a capabilities document, object of class SosContents.

Details

This document provides clients with service metadata about a specific service instance, including metadata about the tightly-coupled data served.
The portions of the GetCapabilities response document that are defined in the OWS Common specification are not modified for SOS. The sections of the response that are specific for the SOS are the Filter_Capabilities and the Contents section.

Objects from the Class

Objects can be created by calls to the construction function of the form SosCapabilities(...) including the parameter "owsVersion" for the respective version of the service metadata document.

Slots

filterCapabilities: Object of class "SosFilter_CapabilitiesOrNULL" ~~
identification: Object of class "OwsServiceIdentificationOrNULL" ~~
provider: Object of class "OwsServiceProviderOrNULL" ~~
operations: Object of class "OwsOperationsMetadataOrNULL" ~~
contents: Object of class "OwsContentsOrNULL" ~~
version: Object of class "character" ~~
updateSequence: Object of class "character" ~~
owsVersion: Object of class "character" ~~

Extends

Class "OwsCapabilities_1.1.0", directly. Class "OwsCapabilities", by class "OwsCapabilities_1.1.0", distance 2.
Methods
No methods defined with class "SosCapabilities_1.0.0" in the signature.

Author(s)
Daniel Nuest <daniel.nuest@uni-muenster.de>

References
Section 8.2.3 of the SOS specification: Na, A., Priest, M. (Eds.), Sensor Observation Service, Open Geospatial Consortium Inc., OGC 06-009r6, Version: 1.0

See Also
SosFilter_Capabilities, SosContents, OwsCapabilities

Examples
showClass("SosCapabilities_1.0.0")
showClass("SosCapabilities_2.0.0")

SosContents-class  Class and Construction Function of "SosContents"

Description
SosContents models the sos:Contents section in a service metadata document.

Usage
SosContents(observationOfferings)

Arguments
observationOfferings
A list of objects of class SosObservationOffering.

Details
The SosContents section extends the generic ows:Contents elements. It contains the SosObservationOfferings of a Sensor Observation Service.

Objects from the Class
Objects can be created by calls to the construction function in the form SosContents(...).
The virtual class SosContentsOrNULL is used to model optional slots in classes containing SosContents:
No objects may be created from it.
sosConvertString

Slots

observationOfferings: Object of class "list"~~
xml: Object of class "XMLAbstractNode"~~

Extends

Class "OwsContents", directly. Class "SosContentsOrNULL", directly. Class "OwsContentsOrNULL", by class "OwsContents", distance 2.

Methods

show signature(object = "SosContents"):

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

References

See section 8.2.3.2, “Contents Section”, of the SOS specification.

See Also

SosObservationOffering, OwsContents

Examples

showClass("SosContents")
showClass("SosContentsOrNULL")

sosConvertString SOS Conversion functions for Observation Results

Description

These functions are called by the parsers of om:Observation and om:Measurement documents to convert the actual values to the correct classes.

Usage

sosConvertString(x, sos)
sosConvertDouble(x, sos)
sosConvertTime(x, sos)
sosConvertLogical(x, sos)
Arguments

- **x**: The object to be converted.
- **sos**: An object of class SOS, whose settings, like formatting information, can be utilized.

Details

The methods are automatically called from the given SOS’s list of conversion functions. This is either default or can be set manually on creation.

If you want to provide your own conversion functions, follow the example below. Always include the common parameters `x` and `sos`.

There are functions to access the converters of a SOS ([sosDataFieldConverters-methods](#)) and to combine default and your own converters ([SosDataFieldConvertingFunctions](#)).

Value

An object of the respective class converted from the parameter `x`.

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

See Also

- [sosDataFieldConverters-methods](#), [SosDataFieldConvertingFunctions](#)

Examples

```
## Not run:
sos <- SOS_Test()
one <- sosConvertDouble("1", sos)
class(one)

# add conversion rules, also possible to override default ones
myConverters <- SosDataFieldConvertingFunctions(
  "C" = sosConvertDouble,
  "S/m" = sosConvertDouble)
sos <- SOS(url = SosExampleServices()[[2]],
dataFieldConverters = myConverters)

# show converters
sosDataFieldConverters(sos)

## End(Not run)
```
Description

These methods can be seen as convenience functions or shortcuts to regularly used parameters in GetObservation requests to a Sensor Observation Service. They remove some complexity and target the most common cases, but also limit flexibility.

NOTE: Whenever time objects are created from characters, note that functions may use your local timezone. All R time objects are UTC-based. All SOS queries use UTC timestamps. This can be confusing when you query data from an SOS located a different timezone or if the time stamp encoded in the request seems to mismatch your input at first glance. Therefore always check if the time period you retrieve matches what you need, and in general stick to using UTC time.

Usage

sosCreateBBOX(lowLat, lowLon, uppLat, uppLon, srsName, srsDimension = NA_integer_,
axisLabels = NA_character_, uomLabels = NA_character_,
propertyName = sosDefaultSpatialOpPropertyName)
sosCreateBBoxMatrix(lowLat, lowLon, uppLat, uppLon)
sosCreateFeatureOfInterest(objectIDs = list(NA), spatialOps = NULL, bbox = NULL,
srsName = NA_character_)
sosCreateTimeInstant(sos, time, frame = as.character(NA),
calendarEraName = as.character(NA),
indeterminatePosition = as.character(NA))
sosCreateTimePeriod(sos, begin, end, frame = as.character(NA),
calendarEraName = as.character(NA),
indeterminatePosition = as.character(NA),
duration = as.character(NA), timeInterval = NULL)
sosCreateTime(sos, time, operator = sosDefaultTemporalOperator)

Arguments

- **lowLat**: Minimum latitude for bounding box and bounding box matrix.
- **lowLon**: Minimum longitude for bounding box and bounding box matrix.
- **uppLat**: Maximum latitude for bounding box and bounding box matrix.
- **uppLon**: Maximum longitude for bounding box and bounding box matrix.
- **srsName**: Name of the spatial reference system for bounding box, e.g. "urn:ogc:def:crs:EPSG:4326".
- **srsDimension**: Dimensions of the spatial reference system, e.g. ‘2’.
- **axisLabels**: Labels of the axes of a bounding box as an ordered character vector.
- **uomLabels**: Unit of measurement labels as an ordered character vector for the axes in a bounding box, e.g. "deg".
**sosCreate**

**propertyName**  The spatial property name for the bounding box, e.g. “urn:ogc:data:location”

**objectIDs**  Identifiers of a feature of interest list.

**spatialOps**  An object of class `OgcSpatialOps-class` which is inserted into the feature of interest element.

**bbox**  Shortcut to add a feature of interest with a `GmlEnvelope-class`, object must be a matrix as created by `sosCreateBBoxMatrix(...)`.

**time**  Object of class "GmlTimeGeometricPrimitive" for `sosCreateEventTimeList`, or an object of class `POSIXt` for `sosCreateTimePeriod`, or an object of class character for `sosCreateTime`. Note that creating timestamps with `as.POSIXct(...)` will use your local timezone.

**operator**  The operator to be used for the time in `sosCreateEventTimeList`, e.g. “TM_During”.

**sos**  An object of class `SOS-class` for which the element is created. The SOS might for example be required for formatting settings.

**frame**  Provides a URI reference that identifies a description of the reference system.

**calendarEraName**  The name of the calendar era.

**begin**  Object of class `POSIXt`. Note that creating timestamps with `as.POSIXct(...)` will use your local timezone.

**end**  Object of class `POSIXt`. Note that creating timestamps with `as.POSIXct(...)` will use your local timezone.

**indeterminatePosition**  Inexact temporal positions may be expressed using the optional `indeterminatePosition` parameter. This takes one of the following values: after, before, now, unknown.

**duration**  Duration of an interval using ISO 8601 syntax for temporal length.

**timeInterval**  An object of class "GmlTimeIntervalOrNULL" to be used in a `GmlTimePeriod-class`.

**Value**

An object of the respective class, or a list in case of `sosEventTimeList`.

**Methods**

signature(time = "GmlTimeGeometricPrimitive")  Create sos:time based on the given `GmlTimeGeometricPrimitive`.

signature(sos = "SOS", time = "POSIXt")  Create sos:time with time instant based on the given time.

signature(sos = "SOS", begin = "POSIXt", end = "POSIXt")  Create sos:time with time interval based on the given begin and end times.

**Author(s)**

Daniel Nuest <daniel.nuest@uni-muenster.de>
See Also

These methods create object of the following classes: \texttt{GmlTimeInstant-class}, \texttt{GmlTimePeriod-class}, \texttt{SosEventTime-class}, \texttt{SosFeatureOfInterest-class}, \texttt{OgcBBOX-class}, \texttt{matrix-class}.

Examples

```r
# create a feature of interest based on identifiers
foiIDs <- list("urn:ogc:object:feature:1", "urn:ogc:object:feature:2")
foiObj <- sosCreateFeatureOfInterest(objectIDs = foiIDs[1:2])
print(foiObj)

# create a bounding box matrix and use it to create a spatial feature of interest
bboxMatrix <- sosCreateBBoxMatrix(lowLat = 50.0, lowLon = 7.0,
                                  uppLat = 53.0, uppLon = 10.0)
foiBBox <- sosCreateFeatureOfInterest(bbox = bboxMatrix,
                                       srsName = "urn:ogc:def:crs:EPSG:6.8:4326")
print(foiBBox)

# create a foi with a bounding box
bbox <- sosCreateBBOX(lowLat = 50.0, lowLon = 7.0, uppLat = 53.0, uppLon = 10.0,
                      srsName = "urn:ogc:def:crs:EPSG:6.8:4326",
                      srsDimension = as.integer(2), axisLabels = "lat,lon",
                      uomLabels = "deg,deg", propertyName = "bboxName")
foiBBox2 <- sosCreateFeatureOfInterest(spatialOps = bbox)
print(foiBBox2)

last.period <- sosCreateTimePeriod(sos = SOS_Test(),
begin = (Sys.time() - 3600 * 24 * 7), end = Sys.time())

period <- sosCreateTimePeriod(sos = SOS_Test(),
begin = as.POSIXct("2010/01/01"), end = as.POSIXct("2010/01/07"))
eventTime <- sosCreateEventTimeList(period)
sosCreateTime(sos = SOS_Test(), time = "2007-07-07 07:00::2008-08-08 08:00")
sosCreateTime(sos = SOS_Test(), time = "2007-07-07 07:00/2010-10-10 10:00")
sosCreateTime(sos = SOS_Test(), time = "::2007-08-05")
sosCreateTime(sos = SOS_Test(), time = "2007-08-05/"")
```

\begin{center}
\textbf{SosEventTime} \hspace{1cm} \textit{Classes and Construction Functions for sos:eventTime elements.}
\end{center}

\textbf{Description}

Temporal query parameters for GetObservation requests.

\textbf{Usage}

```r
SosEventTime(temporalOps)
```
SosEventTime

Arguments

temporalOps  An object of class OgcBinaryTemporalOp-class to be wrapped by the sos:eventTime element.

Details

Specifies the time period(s) for which observations are requested. This allows a client to request observations from a specific instant, multiple instances or periods of time in the past, present and future. The supported range is listed in the selected offering capabilities. The objects of these classes are used in the GetObservation (parameter in GetObservation).

A typical example in a POST request:

```xml
<eventTime>
  <ogc:TM_During>
    <ogc:PropertyName>om:samplingTime</ogc:PropertyName>
    <gml:TimePeriod>
      <gml:beginPosition>2006-11-05T17:18:58.000-06:00</gml:beginPosition>
      <gml:endPosition>2006-11-05T21:18:59.000-06:00</gml:endPosition>
    </gml:TimePeriod>
  </ogc:TM_During>
</eventTime>
```

In GET binding (SosBindings) the eventTime is simply omitted for getting the latest observation. It is recommended to use the creation functions as shown in the examples.

Objects from the Classes

Objects can be created by calls to the construction functions of the form SosEventTime(...).

Slots

temporalOps: Object of class "OgcBinaryTemporalOp" for SosEventTime, the temporal operand to be inserted into the event time, or an object of class "character" for SosEventTimeLatest.

Methods

codeKVP  signature(obj = "SosEventTime", sos = "SOS"): Encode the given object as a key-value pair.

codeXML  signature(obj = "SosEventTime", sos = "SOS"): Encode the given object as XML.

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

References

See SOS specification, Table 4: “Parameters of GetObservation Request”.

See Also

See also SosGetObservation-class, sosCreateEventTimeList-methods.
Examples

showClass("SosEventTime")

# create SosEventTime for all times after the given time stamp
timePos <- GmlTimePosition(as.POSIXct("2010-01-01 12:00"))
tOps <- TM_After(time = GmlTimeInstant(timePosition = timePos))
time1 <- SosEventTime(tOps)

# encode it as XML
encodeXML(time1, sos = SOS_Test())

# encode it as KVP
encodeKVP(time1, sos = SOS_Test())

SosFeatureOfInterest-class

Class and Construction Function for "SosFeatureOfInterest"

Description

Element in a GetObservation request to a Sensor Observation service to constrain the observations to be returned regarding the observed feature.

Usage

SosFeatureOfInterest(objectIDs = list(NA), spatialOps = NULL)

Arguments

  objectIDs A list of character identifiers of features in a SOS.
  spatialOps An object of class OgcSpatialOps for spatial filtering.

Details

Specifies the feature for which observations are requested. This can either be represented by a reference to a feature ID advertised in the capabilities document or can be a spatial constraint.

Objects from the Class

Objects can be created by calls to the construction function of the form SosFeatureOfInterest(...).

SosFeatureOfInterestOrNULL is a virtual class to model optional slots of containing elements: No objects may be created from them.

Slots

  objectIDs: Object of class "list": Identifiers of features of interest.
  spatialOps: Object of class "OgcSpatialOpsOrNULL": A spatial filtering of the result.
SosFilter_Capabilities-class

Extends

Class "SosFeatureOfInterestOrNULL", directly.

Methods

encodeXML signature(obj = "SosFeatureOfInterest", sos = "ANY"): Convert the object to a XML representation.

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

References

See section 8.4.2 of the SOS specification: Na, A., Priest, M. (Eds.), Sensor Observation Service, Open Geospatial Consortium Inc., OGC 06-009r6, Version: 1.0

See Also

See also SosGetObservation, and the convenience creation function sosCreateFeatureOfInterest-methods.

Examples

showClass("SosFeatureOfInterest")
showClass("SosFeatureOfInterestOrNULL")

SosFilter_Capabilities-class

Classes and Construction Functions for "SosFilter_Capabilities" Elements

Description

Additional section in the service metadata document of a Sensor Observation Service, which contains information about the supported filters.

Usage

SosFilter_Capabilities(spatial = list(NA_character_), temporal = list(NA_character_), scalar = list(NA_character_), id = list(NA_character_))

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spatial</td>
<td>A character list of names of available spatial filters.</td>
</tr>
<tr>
<td>temporal</td>
<td>A character list of names of available temporal filters.</td>
</tr>
<tr>
<td>scalar</td>
<td>A character list of names of available scalar filters.</td>
</tr>
<tr>
<td>id</td>
<td>A character list of names of available filters on identifiers.</td>
</tr>
</tbody>
</table>
Details

The FilterCapabilities section is used to indicate what types of query parameters are supported by the service. These capabilities refer to the parameters of the GetObservation operation which is the only operation that includes OGC filter-like expressions.

Objects from the Class

Objects can be created by calls of the form `new("SosFilter_Capabilities",...)`.  
\texttt{SosFilter\_CapabilitiesOrNULL} is virtual class: No objects may be created from it.

Slots

- \texttt{spatial}: Object of class "list" with character strings for names of spatial filters.  
- \texttt{temporal}: Object of class "list" with character strings for names of temporal filters.  
- \texttt{scalar}: Object of class "list" with character strings for names of scalar filters.  
- \texttt{id}: Object of class "list" with character strings for names of ID filters.

Extends

Class "\texttt{SosFilter\_CapabilitiesOrNULL}", directly.

Methods

\texttt{show} signature(object = "SosFilter\_Capabilities"): ...

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

References

See section 8.2.3.1, “FilterCapabilities Section”, the SOS specification: Na, A., Priest, M. (Eds.), Sensor Observation Service, Open Geospatial Consortium Inc., OGC 06-009r6, Version: 1.0

See Also

\texttt{SosCapabilities}

Examples

```r
showClass("SosFilter\_Capabilities")
showClass("SosFilter\_CapabilitiesOrNULL")
```
SosGetDataAvailability_1.0.0-class

Class and construction function for "GetDataAvailability" operation

Description

See SOS 2.0 Hydrology profile specification, OGC 14-004r1, section 7.4, requirement 12

Objects from the Class

Objects can be created by calls to construction functions:
DataAvailabilityMember(...)
SosGetDataAvailability_1.0.0(...)

Slots

- procedure: Object of class "character"
- observedProperty: Object of class "character"
- featureOfInterest: Object of class "character"
- phenomenonTime: Object of class "GmlTimePeriod"
- procedures: Object of class "list"
- observedProperties: Object of class "list"
- featuresOfInterest: Object of class "list"
- offerings: Object of class "list"
- service: Object of class "character"
- request: Object of class "character"
- version: Object of class "character"

Methods

- encodeRequestKVP signature(obj = "SosGetDataAvailability_1.0.0"): encode object as URL request string.
- sosName signature(obj = "SosGetDataAvailability_1.0.0"): access to a name (for human display) of the object.
- toString signature(x = "SosGetDataAvailability_1.0.0"): ...

Extends

Class SosGetDataAvailability_1.0.0 extends "OwsServiceOperation", directly.

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>
References

OGC 14-004r1, section 7.4, requirement 12

Examples

showClass("DataAvailabilityMember")
showClass("SosGetDataAvailability_1.0.0")

SosGetFeatureOfInterest_2.0.0-class

Class "SosGetFeatureOfInterest_2.0.0"

Description

Representation of a GetFeatureOfInterest operation request that needs to be sent to a Sensor Observation Service to retrieve the features of interest, i.e. the real world features that are observed and for which observations are taken.

Objects from the Class

Objects can be created by calls of the form new("SosGetFeatureOfInterest_2.0.0",...).

Slots

featureOfInterest: Object of class "character"
service: Object of class "character"
request: Object of class "character"
version: Object of class "character"

Extends

Class "OwsServiceOperation", directly.

Methods

encodeRequestKVP signature(obj = "SosGetFeatureOfInterest_2.0.0"): ...
sosName signature(obj = "SosGetFeatureOfInterest_2.0.0"): ...

Examples

showClass("SosGetFeatureOfInterest_2.0.0")
Methods for Function `sosObservableProperties` in Package `sos4R`

**Description**

Methods for function `sosObservableProperties` in package `sos4R`. The function allows to retrieve observable properties, e.g. air temperature, wind speed, etc., listed in the capabilities of a Sensor Observation Service and other classes.

**Methods**

- `signature(obj = "list")` List of objects to retrieve properties from.
- `signature(obj = "OmObservation")` Get observable properties from an object of class `OmObservation`.
- `signature(obj = "OmObservationCollection")` Get observable properties from an object of class `OmObservationCollection`, namely the items in the collection.
- `signature(obj = "SOS")` Get observable properties for a whole connection to an SOS.
- `signature(obj = "SosObservationOffering_2.0.0")` Get observable properties from an object of class `SosObservationOffering_2.0.0`, needed internally for getting properties for a whole SOS.
- `signature(obj = "SweCompositePhenomenon")` Get observable properties from an object of class `SweCompositePhenomenon`.
- `signature(obj = "SwePhenomenonProperty")` Get observable properties from an object of class `SwePhenomenonProperty`.

**SosObservationOffering-class**

*Classes and Related Functions for "SosObservationOffering"*

**Description**

`SosObservationOfferings` collect all metadata about a specific offering in a Sensor Observation Service.

**Usage**

```r
SosObservationOffering(id, name = as.character(NA), time, procedure, observedProperty, featureOfInterest, responseFormat, intendedApplication = as.character(NA), resultModel = as.character(NA), responseMode = as.character(NA), boundedBy = list())
```
Arguments

- **boundedBy**: See the corresponding slot description.
- **featureOfInterest**: See the corresponding slot description.
- **id**: See the corresponding slot description.
- **intendedApplication**: See the corresponding slot description.
- **name**: See the corresponding slot description.
- **observedProperty**: See the corresponding slot description.
- **procedure**: See the corresponding slot description.
- **responseFormat**: See the corresponding slot description.
- **responseMode**: See the corresponding slot description.
- **resultModel**: See the corresponding slot description.
- **time**: See the corresponding slot description.

Details

ObservationOffering provides a mechanism for factoring groups of related observations within a single service instance. A functionally equivalent outcome could be obtained by factoring between different service instances.

Value

The construction functions return an object of the respective class, e.g. *SosObservationOffering*.

Objects from the Class

Objects can be created by calls to the construction functions of the form *SosObservationOffering(...)*.

Slots

- **boundedBy**: Object of class "list": A bounding box that contains all features in this offering.
- **featureOfInterest**: Object of class "list": Features or feature collections that represent the identifiable object(s) on which the sensor systems are making observations.
- **id**: Object of class "character": Identifier of an offering.
- **intendedApplication**: Object of class "list": The intended category of use for this offering such as homeland security or natural resource planning.
- **name**: Object of class "character": The name of an offering.
- **observedProperty**: Object of class "list": The observable/phenomenon that can be requested in this offering.
- **procedure**: Object of class "list": A reference to one or more procedures, including sensor systems, instruments, simulators, etc, that supply observations in this offering. The DescribeSensor operation can be called to provide a detailed description of each system.
responseFormat: Object of class "list": MIME type of the data that will be returned as the result of a GetObservation request. This is usually text/xml; subtype="om/0.0.0".

responseMode: Object of class "list": Indicates what modes of response are supported for this offering. The value of resultTemplate is used to retrieve an observation template that will later be used in calls to GetResult. The other options allow results to appear inline in a resultTag (inline), external to the observation element (out-of-band) or as a MIME attachment (attached).

resultModel: Object of class "list": Indicates the namespace-qualified name of the result element that will be included in the document returned from a call to GetObservation for this offering, e.g. "om:Observation" or "om:Measurement".

time: Object of class "GmlTimeGeometricPrimitive": Time period for which observations can be obtained. This supports the advertisement of historical as well as real-time observations.

Methods

sosTime signature(obj = "SosObservationOffering"): Accessor function for the time slot, or to be more precise: the time period for which this offering provides data.

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

References


See Also

SosContents, SosCapabilities

Examples

showClass("SosObservationOffering")
showClass("SosObservationOffering_2.0.0")

# explore offerings of an SOS
mySOS <- SOS(url = "http://sensorweb.demo.52north.org/sensorwebtestbed/service/kvp", binding = "KVP")
offering1 <- sosOfferings(mySOS)[[1]]
sosId(offering1)
sosName(offering1)
sosTime(offering1)
sosBoundedBy(offering1)
**sosRequest-methods**  
*Send Request to SOS*

---

**Description**

This is the main request function for sending and receiving requests respectively documents from a Sensor Observation Service. It’s intended for internal use.

Please use the methods for the SOS operations as long as possible: **getCapabilities-methods**, **describeSensor-methods**, **getObservation-methods**, and **getObservationById-methods**.

**Methods**

```r
signature(sos = "SOS", request = "OwsServiceOperation", verbose = "logical", inspect = "logical")
```

*Method sends the given operation to the given SOS connection. verbose activates extensive debugging to the console. inspect prints only the request and response documents to the console.*

---

**Supported Functions to Access Supported Features of the Current sos4R Implementation**

---

**Description**

These functions can be used to access the supported parameters for a range of settings of a SOS connection.

**Usage**

```r
SosSupportedOperations(version = sos100_version)  
SosSupportedComparisonOperators()  
SosSupportedBindings()  
SosSupportedGeometryOperands()  
SosSupportedResponseFormats()  
SosSupportedResponseModes()  
SosSupportedResultModels()  
SosSupportedSpatialOperators()  
SosSupportedTemporalOperators()  
SosSupportedServiceVersions()
```

**Arguments**

**version**  
The SOS specification version.
Details

Supported features, like connection methods and supported response modes, are accessible by functions starting with "SosSupported". See the examples section for a complete list of these functions.

It is encouraged to rather use these methods than manually set character values for compatibility with future versions, e.g. SosSupportedBindings()[[1]] instead of directly writing "GET".

Value

A list of supported values for the respective parameter.

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

See Also

See Defaults for default values of parameters.

Examples

# The supported operations of the specification
SosSupportedOperations()

# HTTP connection methods supported by this sos4R implementation
SosSupportedBindings()
myBinding <- SosSupportedBindings()[[1]]
myBinding

# Formats, modes and models that can be processed by this implementation
SosSupportedResponseFormats()
SosSupportedResultModels()
SosSupportedResponseModes()

# Operators and operands for filtering in a GetObservation request
SosSupportedTemporalOperators()
SosSupportedSpatialOperators()
SosSupportedGeometryOperands()
SosSupportedComparisonOperators()
Usage

SweCompositePhenomenon(id, name, description = as.character(NA), dimension,
  components, base = NULL)
SwePhenomenon(id, name, description = as.character(NA))
SwePhenomenonProperty(href = as.character(NA), phenomenon = NULL)
SweTextBlock(tokenSeparator, blockSeparator, decimalSeparator, id = as.character(NA))

Arguments

Arguments of the construction functions are as follows.

- **id**
  - The character string to be used for the id attribute (mandatory).
- **name**
  - The character string to be used for the name element (mandatory).
- **description**
  - The character string to be used for the description element.
- **dimension**
  - The dimensions of a composite phenomenon (mandatory).
- **components**
  - The (sub-) components of a composite phenomenon (mandatory).
- **base**
  - The (optional) base element for a composite phenomenon.
- **href**
  - A reference to an (online) object instead of an inline property.
- **phenomenon**
  - The inline phenomenon of a phenomenon property.
- **tokenSeparator**
  - The character to be used as the token separator, often ",".
- **blockSeparator**
  - The character to be used as the block separator, often ";".
- **decimalSeparator**
  - The character to be used as the decimal separator, often ".".

Details

The ...OrNULL classes are used to model optional slots.

Value

The construction functions return an object of the respective class.

Objects from the Class

Objects can be created by calls to the construction functions of the form
SweCompositePhenomenon( ...), SwePhenomenonProperty( ...), and so forth.

The following classes are virtual, no objects may be created from them: SwePhenomenonOrNULL,
SwePhenomenonPropertyOrNULL.

Slots

dimension: Object of class "integer", the value of the dimensions attribute of a composite phenomenon.
components: Object of class "list", the components of a composite phenomenon.
base: Object of class "SwePhenomenonPropertyOrNULL", the base of a composite phenomenon, may be NULL.
id: Object of class "character", the value of the id attribute of a (composite) phenomenon.
name: Object of class "character", the value of the name element of a (composite) phenomenon.
description: Object of class "character", the value of the description elements of a phenomenon.
href: Object of class "character", the value of the href attribute of a phenomenon property which references a phenomenon.
phenomenon: Object of class "SwePhenomenonOrNULL", the actual (inline) phenomenon of a phenomenon property.
tokenSeparator: Object of class "character", the symbol to be used as the token separator in a SweTextBlock, e.g. in the case of "," this would result in attribute1,attribute2.
blockSeparator: Object of class "character", the symbol to be used as the block separator in a SweTextBlock, e.g. in the case of ";" this would result in attribute1_a,attribute2_a;attribute1_b,attribute2_b.
decimalSeparator: Object of class "character", the symbol to be used as the decimal separator in a SweTextBlock, e.g. in the case of "." this would result in attribute1,42.0,attribute3,23.0.

Extends

SweCompositePhenomenon: Class "Swe Phenomenon", directly. Class "Swe PhenomenonOrNULL", by class "Swe Phenomenon", distance 2.
SwePhenomenon: Class "Swe PhenomenonOrNULL", directly.
SwePhenomenonProperty: Class "Swe Phenomenon PropertyOrNULL", directly.

Methods

show signature(object = "SweCompositePhenomenon"): ...
show signature(object = "SwePhenomenon"): ...
show signature(object = "SwePhenomenonProperty"): ...
show signature(object = "SweTextBlock"): ...

Author(s)

Daniel Nuest <daniel.nuest@uni-muenster.de>

References

See section 9, SWE Common XML Encoding and Examples, of Botts, M., Robin, A. (Eds.), OpenGIS(R) Sensor Model Language (SensorML) Implementation Specification, Open Geospatial Consortium Inc., OGC 07-000

Examples

showClass("SweCompositePhenomenon")
showClass("SwePhenomenon")
showClass("SwePhenomenonProperty")
showClass("SwePhenomenonPropertyOrNULL")
showClass("SweTextBlock")
## Not run:
sos <- SOS(...)
obj <- parseX(...)
switchSweNamespace(xml2::xml_ns(obj), sos)

## End(Not run)

---

**SweTextEncoding-class**  
**Class** "SweTextEncoding"

### Description

Representation of a text encoding defined in the OGC SWE Common specification. It defines the token, block, and decimal separators for a text-encoded array of values.

### Objects from the Class

Objects can be created by calls of the form `new("SweTextEncoding",...)`.

### Slots

- `tokenSeparator`: Object of class "character"
- `blockSeparator`: Object of class "character"
- `decimalSeparator`: Object of class "character"
- `id`: Object of class "character"

### Methods

No methods defined with class "SweTextEncoding" in the signature.

### Examples

`showClass("SweTextEncoding")`

---

**TM_Operators**  
**Classes and Construction Methods for Temporal Operator Classes**

### Description

Classes for temporal operators from OpenGIS(R) Filter Encoding used in filters in GetObservation requests.
**TM_Operators**

**Usage**

```r
TM_After(propertyName = sosDefaultTempOpPropertyName, time)
TM_Before(propertyName = sosDefaultTempOpPropertyName, time)
TM_During(propertyName = sosDefaultTempOpPropertyName, time)
TM_Equals(propertyName = sosDefaultTempOpPropertyName, time)
```

**Arguments**

- `propertyName`: The name of the property that is used to wrap the time.
- `time`: A time instant or period to be used as the temporal operand.

**Value**

An object of the respective class, so `TM_After`, `TM_Before`, `TM_During` or `TM_Equals`.

**Objects from the Class**

Objects can be created by calls of the form `new("TM_After",...)`.

**Slots**

- `time`: Object of class "GmlTimeInstant"
- `propertyName`: Object of class "character"

**Extends**


**Methods**

- `encodeXML` signature(obj = "TM_After", sos = "SOS"): ...
- `show` signature(object = "TM_After"): ...
- `encodeXML` signature(obj = "TM_Before", sos = "SOS"): ...
- `show` signature(object = "TM_Before"): ...
- `encodeXML` signature(obj = "TM_During", sos = "SOS"): ...
- `show` signature(object = "TM_During"): ...
- `encodeXML` signature(obj = "TM_Equals", sos = "SOS"): ...
- `show` signature(object = "TM_Equals"): ...

**Author(s)**

Daniel Nuest <daniel.nuest@uni-muenster.de>
References

See the schema file: http://schemas.opengis.net/sos/1.0.0/ogc4sos.xsd.

See Also

SosGetObservation

Examples

```
showClass("TM_After")
showClass("TM_Before")
showClass("TM_During")
showClass("TM_Equals")

## Not run:
# create times to use for operators
t1 <- sosCreateTimeInstant(sos = weathersos, time = Sys.time())
p1 <- sosCreateTimePeriod(sos = weathersos, begin = as.POSIXct("2010-03-01 12:15"),
end = as.POSIXct("2010-03-02 12:15"))

# create temporal operator
afterNow <- TM_After(time = t1)
print(afterNow)
encodeXML(t1, sos)

during <- TM_During(time = p1)
print(during)

## End(Not run)
```

---

WML

Classes, creation, and parsing functions for WaterML

Description

WaterML classes.

WmlTimeseries (abstract)
WmlMeasurementTimeseries
WmlMeasurementTimeseriesMetadata
WmlMeasurementTVP
WmlDefaultTVPMeasurementMetadata
WmlInterpolationType
Objects from the Class

Objects can be created by calls to the creation functions:

- `WmlMeasurementTimeseries(...)`
- `WmlMeasurementTimeseriesMetadata(temporalExtent)`
- `WmlMeasurementTVP(...)`
- `WmlDefaultTVPMeasurementMetadata(...)`
- `WmlInterpolationType(...)`

Slots

- `sampledFeatures`: Object of class "character"
- `id`: Object of class "character"
- `identifier`: Object of class "character"
- `names`: Object of class "character"
- `shape`: Object of class "SamsShape"
- `timeZone`: Object of class "ANY"
- `temporalExtent`: Object of class "GmlPoint"
- `time`: Object of class "POSIXct" in a TVP
- `value`: Object of class "numeric" in a TVP
- `href`: Object of class "character"
- `title`: Object of class "character"
- `interpolationType`: Object of class "WmlInterpolationType"
- `uom`: Object of class "character"
- `metadata`: Object of class "ANY"
- `defaultPointMetadata`: Object of class "ANY"
- `point`: Object of class "ANY" for `WmlTimeseries`, of class "list" for `WmlMeasurementTimeseries`

Extends


Methods

- **sosCoordinates** signature(obj = "SamsSamplingFeature"): access to the coordinates of the feature.
- **sosFeatureIds** signature(obj = "SamsSamplingFeature"): access to the identifier of the feature.
- **sosFeaturesOfInterest** signature(obj = "SamsSamplingFeature"): access the feature itself.
- **sosName** signature(obj = "SamsSamplingFeature"): access a human-readable name of the feature.
Note
Schemas: http://schemas.opengis.net/waterml/

Author(s)
Eike H. Juerrens, Daniel Nuest

References
https://www.ogc.org/standards/WaterML

Examples
showClass("WmlMeasurementTimeseries")
Index

*Topic **XML**
  Constants, 6
  encodeRequestXML-methods, 14
  SosBindings, 60
*Topic **classes**
  DescribeSensor, 10
  GetObservation, 19
  GML, 23
  MonitoringPoint-class, 29
  OGC, 30
  OmMeasurement, 32
  OmObservation-class, 34
  OmObservationCollection, 36
  OmOM_Observation-class, 37
  OWS, 39
  SA, 48
  SAMS, 50
  SML, 54
  SOS, 56
  SosCapabilities, 61
  SosContents-class, 63
  SosEventTime, 68
  SosFeatureOfInterest-class, 70
  SosFilter_Capabilities-class, 71
  SosGetDataAvailability_1.0.0-class, 73
  SosGetFeatureOfInterest_2.0.0-class, 74
  SosObservationOffering-class, 75
  SWE, 79
  SweTextEncoding-class, 82
  TM_Operators, 82
  WML, 84
*Topic **coercion**
  Coercion, 5
*Topic **connection**
  sos4R-package, 3
*Topic **constants**
  Constants, 6
  SosBindings, 60
*Topic **database**
  sos4R-package, 3
*Topic **methods**
  encodeKVP-methods, 13
  encodeRequestKVP-methods, 13
  encodeRequestSOAP-methods, 14
  encodeRequestXML-methods, 14
  encodeXML-methods, 15
  getCapabilities-methods, 16
  GetObservation, 19
  parse, 44
  sosCreate, 66
  sosObservableProperties-methods, 75
  sosRequest-methods, 78
*Topic **misc**
  Defaults, 7
  parse, 44
*Topic **package**
  sos4R-package, 3
*Topic **spatial**
  sos4R-package, 3
*Topic **ts**
  sos4R-package, 3
*Topic **utilities**
  GML, 23
  OGC, 30
  OWS, 39
  sosConvertString, 64
  sosCreate, 66
  Supported, 78
  TM_Operators, 82
*Topic **utilities**
  GetObservation, 19
  [,OmObservationCollection-method
   (OmObservationCollection), 36
  ][,OmObservationCollection,ANY,missing-method
   (OmObservationCollection), 36
encodeRequestKVP, SosGetObservationById-method (encodeRequestKVP-methods), 13
encodeRequestKVP-methods, 13
encodeRequestSOAP, 16
encodeRequestSOAP (encodeRequestSOAP-methods), 14
encodeRequestSOAP, OwsGetCapabilities-method (encodeRequestSOAP-methods), 14
encodeRequestSOAP, SosDescribeSensor-method (encodeRequestSOAP-methods), 14
encodeRequestSOAP, SosGetObservation-method (encodeRequestSOAP-methods), 14
encodeRequestSOAP, SosGetObservationById-method (encodeRequestSOAP-methods), 14
encodeRequestSOAP-methods, 14
encodeRequestXML, 16
encodeRequestXML (encodeRequestXML-methods), 14
encodeRequestXML, OwsGetCapabilities-method (encodeRequestXML-methods), 14
encodeRequestXML, OwsGetCapabilities_1.1.0-method (encodeRequestXML-methods), 14
encodeRequestXML, OwsGetCapabilities_2.0.0-method (encodeRequestXML-methods), 14
encodeRequestXML, SosDescribeSensor-method (encodeRequestXML-methods), 14
encodeRequestXML, SosEventTime, SOS-method (encodeRequestXML-methods), 15
encodeRequestXML, SosFeatureOfInterest, SOS-method (encodeRequestXML-methods), 15
encodeRequestXML, SosGetFeatureOfInterest_2.0.0-method (encodeRequestXML-methods), 15
encodeRequestXML, SosGetFeatureOfInterest_2.0.0-method (encodeRequestXML-methods), 15
encodeRequestXML, SosGetObservation-method (encodeRequestXML-methods), 14
encodeRequestXML, SosGetObservation_2.0.0-method (encodeRequestXML-methods), 14
encodeRequestXML, SosGetObservationById-method (encodeRequestXML-methods), 14
encodeRequestXML, xml_document, SOS-method (encodeXML-methods), 15
encodeRequestXML, xml_node, SOS-method (encodeXML-methods), 15
encodeXML, GmlDirectPosition, SOS-method (encodeXML-methods), 15
encodeXML, GmlEnvelope, SOS-method (encodeXML-methods), 15
encodeXML, GmlLineString, SOS-method (encodeXML-methods), 15
encodeXML, GmlPoint, SOS-method (encodeXML-methods), 15
encodeXML, GmlPointProperty, SOS-method (encodeXML-methods), 15
encodeXML, GmlPolygon, SOS-method (encodeXML-methods), 15
encodeXML, GmlTimeInstant, SOS-method (encodeXML-methods), 15
encodeXML, GmlTimeInstantProperty, SOS-method (encodeXML-methods), 15
encodeXML, GmlTimePeriod, SOS-method (encodeXML-methods), 15
encodeXML, OgcBBOX, SOS-method (encodeXML-methods), 15
encodeXML, OgcComparisonOps, SOS-method (encodeXML-methods), 15
encodeXML, OgcContains, SOS-method (encodeXML-methods), 15
encodeXML, OgcIntersects, SOS-method (encodeXML-methods), 15
encodeXML, OgcOverlaps, SOS-method (encodeXML-methods), 15
encodeXML, POSIXt, SOS-method (encodeXML-methods), 15
encodeXML, SosEventTime, SOS-method (encodeXML-methods), 15
encodeXML, SosFeatureOfInterest, SOS-method (encodeXML-methods), 15
encodeXML, SosGetFeatureOfInterest_2.0.0, SOS-method (encodeXML-methods), 15
encodeXML, SosGetFeatureOfInterest_2.0.0, SOS-method (encodeXML-methods), 15
encodeXML, SosGetObservation, SOS-method (encodeXML-methods), 15
encodeXML, SosGetObservation_2.0.0, SOS-method (encodeXML-methods), 15
encodeXML, TM_After, SOS-method (encodeXML-methods), 15
encodeXML, TM_Before, SOS-method (encodeXML-methods), 15
encodeXML, TM_During, SOS-method (encodeXML-methods), 15
encodeXML, TM_Equals, SOS-method (encodeXML-methods), 15
encodeXML, xml_document, SOS-method (encodeXML-methods), 15
INDEX

GmlFeatureCollection (GML), 23
GmlFeatureCollection-class (GML), 23
gmlFeatureCollectionName (Constants), 6
GmlFeatureMemberName (Constants), 6
GmlFeatureOrGmlFeaturePropertyOrNULL, 29, 50
GmlFeatureOrGmlFeaturePropertyOrNULL-class (GML), 23
GmlFeatureOrNULL, 25, 29, 49, 50
GmlFeatureOrNULL-class (GML), 23
GmlFeatureProperty (GML), 23
GmlFeatureProperty-class (GML), 23
GmlGeometry, 25, 50
GmlGeometry-class (GML), 23
gmlGetReferencedNode (parse), 44
GmlIdentifierName (Constants), 6
GmlIsNodeReferenced (parse), 44
GmlLineString-class (GML), 23
GmlLowerCornerName (Constants), 6
GmlMeasure, 33, 34
GmlMeasure (GML), 23
GmlMeasure-class (GML), 23
gmlNamespace (Constants), 6
gmlNamespacePrefix (Constants), 6
GmlPoint (GML), 23
GmlPoint-class (GML), 23
gmlPointName (Constants), 6
GmlPointOrNULL-class (GML), 23
GmlPointProperty (GML), 23
GmlPointProperty-class (GML), 23
GmlPolygon-class (GML), 23
gmlPosName (Constants), 6
gmlRelatedTimeName (Constants), 6
GmlResultTypeAttributeName (Constants), 6
GmlTimeGeometricPrimitive, 25
GmlTimeGeometricPrimitive-class (GML), 23
GmlTimeInstant (GML), 23
GmlTimeInstant-class (GML), 23
gmlTimeInstantName (Constants), 6
GmlTimeInstantOrNULL, 25
GmlTimeInstantOrNULL-class (GML), 23
GmlTimeInstantProperty (GML), 23
GmlTimeInstantProperty-class (GML), 23
GmlTimeInstantPropertyOrNULL, 25
GmlTimeInstantPropertyOrNULL-class (GML), 23
GmlTimeInterval (GML), 23
GmlTimeInterval-class (GML), 23
gmlTimeIntervalName (Constants), 6
GmlTimeIntervalOrNULL, 25
GmlTimeIntervalOrNULL-class (GML), 23
gmlTimeLengthName (Constants), 6
GmlTimeObject, 25
GmlTimeObject-class (GML), 23
GmlTimeObjectOrNULL, 25
GmlTimeObjectOrNULL-class (GML), 23
GmlTimePeriod (GML), 23
GmlTimePeriod-class (GML), 23
gmlTimePeriodName (Constants), 6
GmlTimePosition (GML), 23
GmlTimePosition-class (GML), 23
gmlTimePositionName (Constants), 6
GmlTimePrimitive, 25
GmlTimePrimitive-class (GML), 23
gmlUpperCornerName (Constants), 6
HTTP (SosBindings), 60
KVP (SosBindings), 60

length, OmObservationCollection-method
(OmObservationCollection), 36
length. OmObservationCollection
(OmObservationCollection), 36

mimeSubtypeOM (Constants), 6
mimeTypeCSV (Constants), 6
mimeTypeOM (Constants), 6
mimeTypeOM20 (Constants), 6
mimeTypeSML (Constants), 6
mimeTypeXML (Constants), 6
MonitoringPoint-class, 29

names.OmMeasurement (OmMeasurement), 32
names.OmObservation
(OmObservation-class), 34
names.OmObservationCollection
(OmObservationCollection), 36

OGC, 30
ogc (OGC), 30
ogcArithmeticOperatorsName (Constants), 6
OgcBBOX (OGC), 30
OgcBBOX-class (OGC), 30
ogcBBOXName (Constants), 6
OgcBinarySpatialOp, 31
OgcBinarySpatialOp (OGC), 30
OgcBinarySpatialOp-class (OGC), 30
OgcBinaryTemporalOp, 83
OgcBinaryTemporalOp (OGC), 30
OgcBinaryTemporalOp-class, 69
OgcBinaryTemporalOp-class (OGC), 30
OgcBinaryTemporalOpOrNULL, 31, 83
OgcBinaryTemporalOpOrNULL-class (OGC), 30
ogcComparisonOpBetweenName (Constants), 6
ogcComparisonOpEqualToName (Constants), 6
ogcComparisonOperatorsName (Constants), 6
ogcComparisonOpGreaterThanName (Constants), 6
ogcComparisonOpGreaterThanOrEqualToName (Constants), 6
ogcComparisonOpIsLikeName (Constants), 6
ogcComparisonOpIsNotEqualTo (Constants), 6
ogcComparisonOpIsNull (Constants), 6
ogcComparisonOpLessThanOrEqualToName (Constants), 6
ogcComparisonOpLessThenName (Constants), 6
OgcComparisonOps (OGC), 30
OgcComparisonOps-class, 20
OgcComparisonOps-class (OGC), 30
OgcComparisonOpsOrXMLOrNULL-class (OGC), 30
OgcContains (OGC), 30
OgcContains-class (OGC), 30
ogcContainsName (Constants), 6
ogcEIDName (Constants), 6
ogcFIDName (Constants), 6
ogcGeometryOperandEnvelopeName (Constants), 6
ogcGeometryOperandLineStringName (Constants), 6
ogcGeometryOperandName (Constants), 6
ogcGeometryOperandPointName (Constants), 6
ogcGeometryOperandPolygonName (Constants), 6
ogcGeometryOperandsName (Constants), 6
ogcIdCapabilities (Constants), 6
ogcIntersectionName (Constants), 6
ogcIntersects (OGC), 30
ogcIntersectsName (Constants), 6
ogcLiteralName (Constants), 6
ogcLogicalOperatorsName (Constants), 6
ogcNamespace (Constants), 6
ogcNamespacePrefix (Constants), 6
OgcOverlaps (OGC), 30
OgcOverlaps-class (OGC), 30
OgcOverlapsName (Constants), 6
ogcPropertyNameName (Constants), 6
ogcScalarCapabilitiesName (Constants), 6
ogcSpatialCapabilitiesName (Constants), 6
ogcSpatialOpBBOXName (Constants), 6
ogcSpatialOpBeyondName (Constants), 6
ogcSpatialOpContainsName (Constants), 6
ogcSpatialOpCoversName (Constants), 6
ogcSpatialOpCrossesName (Constants), 6
ogcSpatialOpDisjointName (Constants), 6
ogcSpatialOpWithinName (Constants), 6
ogcSpatialOpEqualsName (Constants), 6
ogcSpatialOpName (Constants), 6
ogcSpatialOperatorsName (Constants), 6
ogcSpatialOpIntersectsName (Constants), 6
ogcSpatialOpOverlapsName (Constants), 6
OgcSpatialOps, 31
OgcSpatialOps (OGC), 30
OgcSpatialOps-class (OGC), 30
OgcSpatialOpsOrNULL, 31
OgcSpatialOpsOrNULL-class (OGC), 30
ogcSpatialOpTouchesName (Constants), 6
ogcSpatialOpWithinName (Constants), 6
OgcTempOpTMAfterName (Constants), 6
ogcTempOpTMBeforeName (Constants), 6
ogcTempOpTMBeginsName (Constants), 6
ogcTempOpTMbegunByName (Constants), 6
ogcTempOpTMContainsName (Constants), 6
ogcTempOpTMDuringName (Constants), 6
ogcTempOpTMEndedByName (Constants), 6
ogcTempOpTMEndsName (Constants), 6
ogcTempOpTMEqualsName (Constants), 6
ogcTempOpTMMeetsName (Constants), 6
ogcTempOpTMetByName (Constants), 6
ogcTempOpTMOverlappedBy (Constants), 6
OwsGetCapabilities (OWS), 39
OwsGetCapabilities-class (OWS), 39
OwsGetCapabilities_1.1.0, 43
OwsGetCapabilities_1.1.0-class (OWS), 39
OwsGetCapabilities_2.0.0-class (OWS), 39
owsGetName (Constants), 6
owsHTTPName (Constants), 6
owsKeyName (Constants), 6
owsKeywordsName (Constants), 6
owsMaximumValueName (Constants), 6
owsMinimumValueName (Constants), 6
owsNamespace (Constants), 6
owsNamespacePrefix (Constants), 6
OwsOperation (OWS), 39
OwsOperation-class (OWS), 39
owsOperationName (Constants), 6
OwsOperationsMetadata (OWS), 39
OwsOperationsMetadata-class (OWS), 39
owsOutputFormatName (Constants), 6
owsParameterName (Constants), 6
owsProfileName (Constants), 6
owsProviderNameName (Constants), 6
owsProviderSiteName (Constants), 6
OwsRange (OWS), 39
OwsRange-class (OWS), 39
owsRangeName (Constants), 6
owsSectionName (Constants), 6
owsSectionsName (Constants), 6
owsServiceProviderTypeName (Constants), 6
OwsServiceProviderIdentification (OWS), 39
OwsServiceProviderIdentification-class (OWS), 39
owsServiceProviderIdentificationName (Constants), 6
OwsServiceProviderIdentificationOrNULL, 43
OwsServiceProviderIdentificationOrNULL-class (OWS), 39
OwsServiceProviderOperation, 11, 21, 43, 73, 74
OwsServiceProvider-class (OWS), 39
OwsServiceProviderName (OWS), 39
OwsServiceProvider-class (OWS), 39
owsServiceProviderName (Constants), 6
OwsServiceProviderOrNULL, 43
OwsServiceProviderOrNULL-class (OWS), 39
owsServiceType (Constants), 6
owsServiceTypeVersionName (Constants), 6
owsSpacingName (Constants), 6
owsTitleName (Constants), 6
owsValueName (Constants), 6
owsVersionName (Constants), 6
parse, 44
parseCategoryObservation (parse), 44
parseComplexObservation (parse), 44
parseComponent (parse), 44
parseCompositePhenomenon (parse), 44
parseCountObservation (parse), 44
parseCSV (parse), 44
parseDataArray (parse), 44
parseDescribeSensorResponse (parse), 44
parseElementType (parse), 44
parseEncoding (parse), 44
parseEnvelope (parse), 44
parseFeatureCollection (parse), 44
parseField (parse), 44
parseFile (parse), 44
parseFile, SOS_1.0.0, character-method (parse), 44
parseFile, SOS_versioned, character-method (parse), 44
parseFile-method (parse), 44
parseFOM (parse), 44
parseGeometryObservation (parse), 44
parseGetDataAvailabilityResponse (SosGetDataAvailability_1.0.0-class), 73
parseGetFeatureOfInterestResponse (parse), 44
parseGetObservationByIdResponse (parse), 44
parseGetObservationResponse (parse), 44
parseMeasure (parse), 44
parseMeasurement (parse), 44
parseNoParsing (parse), 44
parseObservation (parse), 44
parseObservation_2.0 (parse), 44
parseObservationCollection (parse), 44
parseObservationProperty (parse), 44
parseOM (parse), 44
parseOwsException (parse), 44
parseOwsExceptionReport (parse), 44
parseOwsOperation (parse), 44
parseOwsServiceProvider (parse), 44
parseOwsServiceIdentification (parse), 44
parseOwsServiceRange (parse), 44
parsePhenomenonProperty (parse), 44
parsePoint (parse), 44
parsePosition (parse), 44
parseResult (parse), 44
parseSamplingPoint (parse), 44
parseSamsShape (parse), 44
parseSams200SamplingFeature (SAMS), 50
parseSweCoordinate (parse), 44
parseSweCoordinate-method (parse), 44
parseSweLocation (parse), 44
parseSweLocation-method (parse), 44
parseSwePosition (parse), 44
parseSwePosition-method (parse), 44
parseSwesObservableProperty (parse), 44
parseSweVector (parse), 44
parseSweVector-method (parse), 44
parseTemporalObservation (parse), 44
parseTextBlock (parse), 44
parseTextEncoding (parse), 44
parseTime (parse), 44
parseTimeGeometricPrimitiveFromParent (parse), 44
parseTimeInstant (parse), 44
parseTimeInstantProperty (parse), 44
parseTimeObject (parse), 44
parseTimePeriod (parse), 44
parseTimePosition (parse), 44
parseTruthObservation (parse), 44
parseValues (parse), 44
parseWmlDefaultTVPMeasurementMetadata (WML), 84
parseWmlInterpolationType (WML), 84
parseWmlMeasurementTimeseries (WML), 84
parseWmlMeasurementTimeseriesMetadata (WML), 84
parseWmlMeasurementTVP (WML), 84
parseWmlMonitoringPoint (parse), 44
phenomena, 47
phenomena, SOS_2.0.0-method (phenomena), 47
phenomena-methods (phenomena), 47
plot, SensorML, missing-method (SML), 54
plot, SOS, missing-method (SOS), 56
plot, SosObservationOffering, missing-method (SOS), 56
plot, SensorML (SML), 54
plot, SOS (SOS), 56
plot, SosObservationOffering (SOS), 56
POST (SosBindings), 60
POX (SosBindings), 60
print, DataAvailabilityMember-method (SosGetDataAvailability_1.0.0-class), 73
print, GmlDirectPosition-method (GML), 23
print, GmlEnvelope-method (GML), 23
print, GmlFeatureCollection-method (GML), 23
print, GmlFeatureProperty-method (GML), 23
print, GmlGeometry-method (GML), 23
print, GmlMeasure-method (GML), 23
print, GmlPoint-method (GML), 23
print, GmlPointProperty-method (GML), 23
print, GmlTimeInstant-method (GML), 23
print, GmlTimeInstantProperty-method (GML), 23
print, GmlTimeInterval-method (GML), 23
print, OgcBBOX-method (OGC), 30
print, OgcContains-method (OGC), 30
print, OgcIntersects-method (OGC), 30
print, OgcOverlaps-method (OGC), 30
print, OmMeasurement-method (OmMeasurement), 32
print, OmObservation-method (OmObservation-class), 34
print, OmObservationCollection-method (OmObservationCollection), 36
print, OmObservationProperty-method (OmObservation-class), 34
print, OwsCapabilities-method (OWS), 39
print, OwsCapabilities_1.1.0-method (OwsCapabilities), 34
print, SosCapabilities-method (SosCapabilities), 34
print, SosCapabilities100-method (SosCapabilities), 34
print, SosCapabilities200-method (SosCapabilities), 34
print, SosCapabilities_1.0.0-method (SosCapabilities), 34
print, SosCapabilities_1.1.0-method (SosCapabilities), 34
(OWS), 39
print,OwsCapabilities_2.0.0-method (OWS), 39
print,OwsContents-method (OWS), 39
print,OwsException-method (OWS), 39
print,OwsExceptionReport-method (OWS), 39
print,OwsGetCapabilities-method (OWS), 39
print,OwsGetCapabilities_1.1.0-method (OWS), 39
print,OwsGetCapabilities_2.0.0-method (OWS), 39
print,OwsOperation-method (OWS), 39
print,OwsOperationsMetadata-method (OWS), 39
print,OwsRange-method (OWS), 39
print,OwsServiceIdentification-method (OWS), 39
print,OwsServiceOperation-method (OWS), 39
print,OwsServiceProvider-method (OWS), 39
print,SamsSamplingFeature-method (SAMS), 50
print,SamsShape-method (SAMS), 50
print,SaSamplingPoint-method (SA), 48
print,SaSamplingSurface-method (SA), 48
print,SensorML-method (SWE), 54
print,SOS-method (SWE), 56
print,SOS_1.0.0-method (SWE), 56
print,SOS_2.0.0-method (SWE), 56
print,SosContents-method (SWE), 63
print,SosDescribeSensor-method (SWE), 10
print,SosEventTime-method (SWE), 68
print,SosFeatureOfInterest-method (SWE), 70
print,SosFilter_Capabilities-method (SWE), 71
print,SosGetObservation-method (SWE), 19
print,SosGetObservationByld-method (SWE), 19
print,SosObservationOffering-method (SWE), 75
print,SosObservationOffering_2.0.0-method (SWE), 75
print,SweCompositePhenomenon-method (SWE), 79
print,SwePhenomenon-method (SWE), 79
print,SwePhenomenonProperty-method (SWE), 79
print,SweTextBlock-method (SWE), 79
print,SweTextEncoding-method (SWE), 79
print,TM_After-method (SWE), 82
print,TM_Before-method (SWE), 82
print,TM_During-method (SWE), 82
print,TM_Equals-method (SWE), 82
print,WmlMonitoringPoint-method (SWE), 29
print.summary.GmlTimePeriod (GML), 23
print.summary.OmObservation (OmObservation), 34
print.summary.OmObservationCollection (OmObservationCollection), 36
print.summary.SosObservationOffering (SOS), 56
print.summary.SosObservationOffering_2.0.0 (SOS), 75
release_questions (sos4R-package), 3
SA, 48
sa (SA), 48
samNamespace (Constants), 6
samNamespacePrefix (Constants), 6
sampling features (SA), 48
SAMS, 50
samSampledFeatureName (Constants), 6
samsNamespace (Constants), 6
samsNamespacePrefix (Constants), 6
SamsSamplingFeature (SAMS), 50
SamsSamplingFeature-class (SAMS), 50
samsSamplingFeatureName (Constants), 6
SamsShape (SAMS), 50
SamsShape-class (SAMS), 50
samsShapeName (Constants), 6
SamsShapeOrNULL (SAMS), 50
SamsShapeOrNULL-class (SAMS), 50
samTypeName (Constants), 6
saNamespace (Constants), 6
saNamespacePrefix (Constants), 6
saPositionName (Constants), 6
saSampledFeatureName (Constants), 6
SaSamplingPoint (SA), 48
SaSamplingPoint-class (SA), 48
saSamplingPointName (Constants), 6
SaSamplingSurface (SA), 48
saSamplingSurface (Constants), 6
SaSamplingSurface-class (SA), 48
saSamplingTimeName (Constants), 6
SensorML, 12
SensorML (SML), 54
SensorML-class (SML), 54
show, DataAvailabilityMember-method
    (SosGetDataAvailability_1.0.0-class), 73
show, GmlDirectPosition-method (GML), 23
show, GmlEnvelope-method (GML), 23
show, GmlFeatureCollection-method (GML), 23
show, GmlFeatureProperty-method (GML), 23
show, GmlGeometry-method (GML), 23
show, GmlMeasure-method (GML), 23
show, GmlPoint-method (GML), 23
show, GmlPointProperty-method (GML), 23
show, GmlTimeInstant-method (GML), 23
show, GmlTimeInterval-method (GML), 23
show, GmlTimePeriod-method (GML), 23
show, GmlTimePosition-method (GML), 23
show, Ogcbbox-method (OGC), 30
show, OgcContains-method (OGC), 30
show, OgcIntersects-method (OGC), 30
show, OgcOverlaps-method (OGC), 30
show, OmMeasurement-method
    (OmMeasurement), 32
show, OmObservation-method
    (OmObservation-class), 34
show, OmObservationCollection-method
    (OmObservationCollection), 36
show, OmObservationProperty-method
    (OmObservation-class), 34
show, OwsCapabilities-method (OWS), 39
show, OwsCapabilities_1.1.0-method
    (OWS), 39
show, OwsCapabilities_2.0.0-method
    (OWS), 39
show, OwsContents-method (OWS), 39
show, OwsException-method (OWS), 39
show, OwsExceptionReport-method (OWS), 39
show, OwsGetCapabilities-method (OWS), 39
show, OwsGetCapabilities_1.1.0-method
    (OWS), 39
show, OwsGetCapabilities_2.0.0-method
    (OWS), 39
show, OwsOperation-method (OWS), 39
show, OwsOperationsMetadata-method
    (OWS), 39
show, OwsRange-method (OWS), 39
show, OwsServiceIdentification-method
    (OWS), 39
show, OwsServiceOperation-method (OWS), 39
show, OwsServiceProvider-method (OWS), 39
show, SamsSamplingFeature-method (SAMS), 50
show, SamsShape-method (SAMS), 50
show, SaSamplingPoint-method (SA), 48
show, SaSamplingSurface-method (SA), 48
show, SensorML-method (SML), 54
show, SOS-method (SOS), 56
show, SOS_1.0.0-method (SOS), 56
show, SOS_2.0.0-method (SOS), 56
show, SosCapabilities-method
    (SosCapabilities-class), 63
show, SosDescribeSensor-method
    (DescribeSensor), 10
show, SosEventTime-method
    (SosEventTime), 68
show, SosFeatureOfInterest-method
    (SosFeatureOfInterest-class), 70
show, SosFilterCapabilities-method
    (SosFilterCapabilities-class), 71
show, SosGetObservation-method
    (GetObservation), 19
show, SosGetObservationById-method
    (GetObservation), 19
show, SosObservationOffering-method
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Method/Class</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>show</td>
<td>SosObservationOffering-class</td>
<td>75</td>
</tr>
<tr>
<td>show</td>
<td>SosObservationOffering_2.0.0-method</td>
<td>75</td>
</tr>
<tr>
<td>show</td>
<td>SweCompositePhenomenon-method</td>
<td>79</td>
</tr>
<tr>
<td>show</td>
<td>SwePhenomenon-method (SWE)</td>
<td>79</td>
</tr>
<tr>
<td>show</td>
<td>SwePhenomenonProperty-method (SWE)</td>
<td>79</td>
</tr>
<tr>
<td>show</td>
<td>SweTextBlock-method (SWE)</td>
<td>79</td>
</tr>
<tr>
<td>show</td>
<td>TM_Before-method (TM_Operators)</td>
<td>82</td>
</tr>
<tr>
<td>show</td>
<td>TM_After-method (TM_Operators)</td>
<td>82</td>
</tr>
<tr>
<td>show</td>
<td>TM_Equals-method (TM_Operators)</td>
<td>82</td>
</tr>
<tr>
<td>show</td>
<td>WMlMonitoringPoint-method (MonitoringPoint-class)</td>
<td>29</td>
</tr>
<tr>
<td>siteList</td>
<td>51</td>
<td></td>
</tr>
<tr>
<td>siteList</td>
<td>SOS_2.0.0-method (siteList)</td>
<td>51</td>
</tr>
<tr>
<td>siteList-methods</td>
<td>siteList</td>
<td>51</td>
</tr>
<tr>
<td>sites</td>
<td>53</td>
<td></td>
</tr>
<tr>
<td>sites</td>
<td>SOS_2.0.0-method (sites)</td>
<td>53</td>
</tr>
<tr>
<td>sites-methods</td>
<td>sites</td>
<td>53</td>
</tr>
<tr>
<td>SML</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>sml</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>smlNamespace</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>smlSensorMLName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>SOAP/SosBindings</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>SOS</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>sos100_version</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos100Namespace</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos100NamespacePrefix</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200_version</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200ContentsName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200ContentsName (Constants)</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>sos200DefaultGetObsResponseFormat (Defaults)</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>sos200FeatureOfInterestName (Constants)</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>sos200FeatureOfInterestType Name (Constants)</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>sos200FilterCapabilitiesName (Constants)</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>sos200Namespace</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200NamespacePrefix</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200ObservationDataName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200ObservationName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200ObservationOffering ListName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200ObservationOfferingName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200ObservationTypeName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200ObservedAreaName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200PhenomenonTimeName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200ResponseFormatName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200ResponseModeName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200ResultTimeName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200SpatialFilterName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos200TemporalFilterName</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>sos4R</td>
<td>(sos4R-package)</td>
<td>3</td>
</tr>
<tr>
<td>sos4R-package</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>sos4R_caches</td>
<td>(parse)</td>
<td>44</td>
</tr>
<tr>
<td>SOS_1.0.0</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SOS_1.0.0-class</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SOS_2.0.0</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SOS_2.0.0-class</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SOS_Test</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SOS_Test-class</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SOS_versioned</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SOS_versioned-class</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>sosAbstract</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>sosAbstract, OwsServiceIdentification-method (OWS)</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>sosAbstract, SensorML-method</td>
<td>(SML)</td>
<td>54</td>
</tr>
<tr>
<td>sosAbstract, SOS-method</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>sosAbstract-method</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>sosAllNamespaces</td>
<td>(Constants)</td>
<td>6</td>
</tr>
<tr>
<td>SosBinding</td>
<td>(SosBindings)</td>
<td>60</td>
</tr>
<tr>
<td>SosBinding</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SosBinding, SOS-method</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SosBinding, SOS_1.0.0-method</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SosBinding, SOS_2.0.0-method</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SosBinding-methods</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>SosBindings</td>
<td>13–15, 20, 21, 60, 69</td>
<td></td>
</tr>
<tr>
<td>sosBoundedBy</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>sosBoundedBy, listing</td>
<td>(SOS)</td>
<td>56</td>
</tr>
<tr>
<td>sosBoundedBy, OmObservationCollection-method (OmObservationCollection)</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>sosBoundedBy, SensorML-method</td>
<td>(SML)</td>
<td>54</td>
</tr>
<tr>
<td>sosBoundedBy, SosObservationOffering-method (SosObservationOffering-class)</td>
<td>75</td>
<td></td>
</tr>
<tr>
<td>sosBoundedBy, SosObservationOffering_2.0.0-method</td>
<td>75</td>
<td></td>
</tr>
</tbody>
</table>
sosCreateTimeInstant, SOS, POSIXt-method
  (sosCreate), 66
sosCreateTimeInstant-methods
  (sosCreate), 66
sosCreateTimePeriod (sosCreate), 66
sosCreateTimePeriod, SOS, POSIXt, POSIXt-method
  (sosCreate), 66
sosCreateTimePeriod-methods
  (sosCreate), 66
sosDataFieldConverters (SOS), 56
sosDataFieldConverters, SOS-method
  (SOS), 56
sosDataFieldConverters, SOS_Test-method
  (SOS), 56
sosDataFieldConverters-methods
  (SOS), 56
SosDataFieldConvertingFunctions, 57, 65
SosDataFieldConvertingFunctions
  (Defaults), 7
sosDefault (Defaults), 7
SosDefaultBinding (Defaults), 7
sosDefaultCharacterEncoding (Defaults), 7
sosDefaultColorPalette (Defaults), 7
sosDefaultColumnNameFeatureIdentifier
  (Defaults), 7
sosDefaultColumnNameLat (Defaults), 7
sosDefaultColumnNameLon (Defaults), 7
sosDefaultColumnNameSRS (Defaults), 7
SosDefaultDCPs (Defaults), 7
sosDefaultDescribeSensorOutputFormat
  (Defaults), 7
sosDefaultFilenameTimeFormat
  (Defaults), 7
sosDefaultGetCapAcceptFormats
  (Defaults), 7
sosDefaultGetCapOwsVersion (Defaults), 7
sosDefaultGetCapSections (Defaults), 7
sosDefaultGetObsResponseFormat
  (Defaults), 7
SosDefaultParsingOptions (Defaults), 7
sosDefaultReferenceFrameSensorDescription
  (Defaults), 7
SosDefaults (Defaults), 7
SosDefaults2 (Defaults), 7
sosDefaultSpatialOpPropertyReference
  (Defaults), 7
sosDefaultTempOpPropertyReference
  (Defaults), 7
sosDescribeFeatureTypeName (Constants), 6
sosDescribeObservationTypeName
  (Constants), 6
sosDescribeResultModelName (Constants), 6
SosDescribeSensor (DescribeSensor), 10
SosDescribeSensor-class, 11
SosDescribeSensor-class
  (DescribeSensor), 10
SosDisabledParsers (Defaults), 7
sosEncoders (SOS), 56
sosEncoders, SOS-method (SOS), 56
SosEncodingFunctions (Defaults), 7
SosEventTime, 68
SosEventTime-class, 20
SosEventTime-class (SosEventTime), 68
sosEventTimeName (Constants), 6
sosExceptionCodeMeaning (SOS), 56
sosExceptionCodeMeaning, character-method
  (SOS), 56
sosFeatureIds (SOS), 56
sosFeatureIds, DataAvailabilityMember-method
  (SosGetDataAvailability_1.0.0-class), 73
sosFeatureIds, GmlFeatureCollection-method
  (GML), 23
sosFeatureIds, GmlFeatureProperty-method
  (GML), 23
sosFeatureIds, GmlTimeObject-method
  (GML), 23
sosFeatureIds, list-method (SOS), 56
sosFeatureIds, OmMeasure-method
  (OmMeasurement), 32
sosFeatureIds, OmObservation-method
  (OmObservation-class), 34
sosFeatureIds, OmObservationCollection-method
  (OmObservationCollection), 36
sosFeatureIds, OmOM_Observation-method
  (OmOM_Observation-class), 37
sosFeatureIds, SamsSamplingFeature-method
  (SAMS), 50
sosFeatureIds, SaSamplingPoint-method
INDEX

(SA), 48
sosFeatureIds,WmlMonitoringPoint-method (WML), 84
sosFeatureIds-method (SOS), 56
SosFeatureOfInterest, 20
SosFeatureOfInterest (SosFeatureOfInterest-class), 70
SosFeatureOfInterest-class, 70
sosFeatureOfInterestName (Constants), 6
sosFeatureOfInterestOrNULL (SosFeatureOfInterest-class), 70
sosFeatureOfInterestTypeName (Constants), 6
sosFeaturesOfInterest (SOS), 56
sosFeaturesOfInterest,DataAvailabilityMember-method (SosGetDataAvailability_1.0.0-class), 73
sosFeaturesOfInterest,GmlFeatureCollection-method (GML), 23
sosFeaturesOfInterest,GmlFeatureProperty-method (GML), 23
sosFeaturesOfInterest,list-method (SOS), 56
sosFeaturesOfInterest,OmMeasurement-method (OmMeasurement), 32
sosFeaturesOfInterest,OmObservation-method (OmObservation-class), 34
sosFeaturesOfInterest,OmObservationCollectionsosGetCRS,OmObservationCollection-method (OmObservationCollection), 36
sosFeaturesOfInterest,OmObservationCollectionsosGetCRS,OmObservationCollection-method (OmObservationCollection), 36
sosFeaturesOfInterest,OmOM_Observation-method (OmOM_Observation-class), 37
sosFeaturesOfInterest,SamsSamplingFeature-method (SAMS), 50
sosFeaturesOfInterest,SaSamplingPoint-method (SA), 48
sosFeaturesOfInterest,SOS,character-method (SOS), 56
sosFeaturesOfInterest,SOS-method (SOS), 56
sosFeaturesOfInterest,SOS_2.0.0-method (SOS), 56
sosFeaturesOfInterest,SosObservationOffering-method (SOS), 56
sosFeaturesOfInterest-methods (SOS), 56
SosFilter_Capabilities, 63
SosFilter_Capabilities (SosFilter_Capabilities-class), 71
sosFilter_Capabilities (SOS), 56
sosFilter_Capabilities, SOS-method (SOS), 56
SosFilter_Capabilities-class, 71
sosFilter_Capabilities-methods (SOS), 56
SosFilter_CapabilitiesOrNULL, 72
SosFilter_CapabilitiesOrNULL-class (SosFilter_Capabilities-class), 71
sosFilterCapabilitiesName (Constants), 6
sosGDAMemberName (Constants), 6
sosGetCapabilitiesName (Constants), 6
sosGetCRS,character-method (SOS), 56
sosGetCRS,GmlDirectPosition-method (GML), 23
sosGetCRS,GmlEnvelope-method (GML), 23
sosGetCRS,GmlPoint-method (GML), 23
sosGetCRS,GmlPointProperty-method (GML), 23
sosGetCRS,list-method (SOS), 56
sosGetCRS,OmMeasurement-method (OmMeasurement), 32
sosGetCRS,OmObservation-method (OmObservation-class), 34
sosGetCRS,OmObservationCollectionsosGetCRS,OmObservationCollection-method (OmObservationCollection), 36
sosGetCRS,OmObservationCollectionsosGetCRS,OmObservationCollection-method (OmObservationCollection), 36
sosGetCRS,OmOM_Observation-method (OmOM_Observation-class), 37
sosGetCRS,SensorML-method (SML), 54
sosGetCRS,SOS-method (SOS), 56
sosGetCRS,SosObservationOffering-method (SOS), 56
sosGetCRS-method (SOS), 56
SosGetDataAvailability_1.0.0-class (SosGetDataAvailability_1.0.0-class), 73
SosGetDataAvailability_1.0.0-class, 73
sosGetDataAvailabilityName (Constants), 6
sosGetDataAvailabilityResponse (Constants), 6
sosGetDCP (SOS), 56
SosGetFeatureOfInterest_2.0.0
sosGetFeatureOfInterest_2.0.0-class, 74
SosGetFeatureOfInterest_2.0.0-class, 18, 74
sosGetFeatureOfInterestName (Constants), 6
sosGetFeatureOfInterestResponseName (Constants), 6
sosGetFeatureOfInterestTimeName (Constants), 6
SosGetObservation, 71, 84
SosGetObservation (GetObservation), 19
SosGetObservation-class, 20
SosGetObservation-class (GetObservation), 19
SosGetObservation_2.0.0 (GetObservation), 19
SosGetObservationById (GetObservation), 19
SosGetObservationById-class, 20
SosGetObservationById-class (GetObservation), 19
sosGetObservationByIdName (Constants), 6
sosGetObservationByIdResponseName (Constants), 6
sosGetObservationResponseName (Constants), 6
sosGetResultName (Constants), 6
sosId (SOS), 56
sosId, GmlFeature-method (GML), 23
sosId, GmlFeatureProperty-method (GML), 23
sosId, list-method (SOS), 56
sosId, SensorML-method (SML), 54
sosId, SosObservationOffering-method (SosObservationOffering-class), 75
sosId, SosObservationOffering_2.0.0-method (SosObservationOffering-class), 75
sosId-method (SOS), 56
sosInsertObservationName (Constants), 6
sosIntendedApplicationName (Constants), 6
sosIsListFieldAvailable (parse), 44
sosKVPParamNameBBOX (Constants), 6
sosKVPParamNameEventTime (Constants), 6
sosKVPParamNameFoi (Constants), 6
sosKVPParamNameNamespaces (Constants), 6
sosKVPParamNameObsProp (Constants), 6
sosKVPParamNameOffering (Constants), 6
sosKVPParamNameProcedure (Constants), 6
sosKVPParamNameRequest (Constants), 6
sosKVPParamNameResponseFormat (Constants), 6
sosKVPParamNameResponseMode (Constants), 6
sosKVPParamNameResultModel (Constants), 6
sosKVPParamNameService (Constants), 6
sosKVPParamNameSpatialFilter (Constants), 6
sosKVPParamNameSrsName (Constants), 6
sosKVPParamNameTemporalFilter (Constants), 6
sosKVPParamNameVersion (Constants), 6
sosName (SOS), 56
sosName, GmlFeatureProperty-method (GML), 23
sosName, list-method (SOS), 56
sosName, OwsGetCapabilities-method (SOS), 56
sosName, OwsOperation-method (SOS), 56
sosName, OwsServiceProvider-method (SOS), 56
sosName, SamsSamplingFeature-method (SAMS), 50
sosName, SensorML-method (SML), 54
sosName, SosDescribeSensor-method (SOS), 56
sosName, SosGetDataAvailability_1.0.0-method (SosGetDataAvailability_1.0.0-class), 73
sosName, SosGetFeatureOfInterest_2.0.0-method (SOS), 56
sosName, SosGetObservation-method (SOS), 56
sosName, SosGetObservation_2.0.0-method (SOS), 56
sosName, SosGetObservationById-method (SOS), 56
sosName, SosObservationOffering-method (SosObservationOffering-class),
sosOfferingIds-methods (SOS), 56
sosOfferingName (Constants), 6
sosOfferings (SOS), 56
sosOfferings, SOS, character-method (SOS), 56
sosOfferings, SOS-method (SOS), 56
sosOffering-methods (SOS), 56
sosOperation (SOS), 56
sosOperation, SOS, character-method (SOS), 56
sosOperation-methods (SOS), 56
sosOperations (SOS), 56
sosOperations, OwsCapabilities-method (SOS), 56
sosOperations, SOS-method (SOS), 56
sosOperations, SosCapabilities_1.0.0-method (SOS), 56
sosOperations, SosCapabilities_2.0.0-method (SOS), 56
sosOperationsMetadata (SOS), 56
sosOperationsMetadata-methods (SOS), 56
sosOperationsMetadata, sosOperationsMethod-method (SOS), 56
sosOutputFormats (SOS), 56
sosOutputFormats, OwsOperation-method (SOS), 56
sosOutputFormats-methods (SOS), 56
sosOutputFormats, sosOutputFormat-method (SOS), 56
sosParsers (SOS), 56
sosParsers-methods (SOS), 56
SosParse-methods (parse), 44
sosParse-methods (parse), 44
sosParse, SOS_1.0.0, character, logical-method (parse), 44
sosParse, SOS_1.0.0, character-method (parse), 44
sosParse-methods (parse), 44
sosParsers (SOS), 56
sosParsers, SOS-method (SOS), 56
sosParsers, SOS_Test-method (SOS), 56
sosParsers-methods (SOS), 56
SosParsingFunctions, 46
SosParsingFunctions (Defaults), 7
sosPhenomenonTimeName (Constants), 6
sosProcedureDescriptionFormat (Constants), 6
sosProcedureDescriptionFormat (SosGetDataAvailability_1.0.0-class), 73
sosProcedureDescriptionFormat (SosGetDataAvailability_2.0.0-class), 73
sosProcedureName (Constants), 6
sosProcedures (SOS), 56
sosProcedures, SosGetDataAvailability-method (SosGetDataAvailability_1.0.0-class), 73
sosProcedures, SosGetDataAvailability-method (SosGetDataAvailability_2.0.0-class), 73
sosProcedures, list-method (SOS), 56
sosProcedures, OmMeasurement-method (OmMeasurement), 32
sosProcedures, OmObservation-method (OmObservation-class), 34
sosProcedures, OmObservationCollection-method (OmObservationCollection), 36
sosProcedures, OmOM_Observation-method (OmOM_Observation-class), 37
sosProcedures, SOS-method (SOS), 56
sosProcedures, SosObservationOffering-method (SOS), 56
sosProcedures, SosObservationOffering_2.0.0-method (SOS), 56
sosProcedures-methods (SOS), 56
sosRegisterSensorName (Constants), 6
sosRequest (sosRequest-methods), 78
sosRequest, SOS_1.0.0, OwsServiceOperation, logical, logical-method (sosRequest-methods), 78
sosRequest, SOS_2.0.0, OwsServiceOperation, logical, logical-method (sosRequest-methods), 78
sosRequest-methods, 78
SosResetParsingFunctions (Defaults), 7
sosResponseFormatName (Constants), 6
sosResponseFormatName (SOS), 56
sosResponseFormatName, OwsOperation-method (SOS), 56
sosResponseFormatName-methods (SOS), 56
sosResponseFormats, SOS-method (SOS), 56
sosResponseFormats-methods (SOS), 56
sosResponseMode (SOS), 56
sosResponseMode, OwsOperation-method (SOS), 56
sosResponseMode-methods (SOS), 56
sosResponseMode, SosObservationOffering-method (SOS), 56
sosResponseMode-methods (SOS), 56
sosResult (OmObservation-class), 34
sosResult, character-method (SOS), 56
sosResult, data.frame-method (SOS), 56
sosResult, list-method (OmObservation-class), 34
sosResult, OmMeasurement-method (OmMeasurement), 32
sosResult,OmObservation-method (OmObservation-class), 34
sosResult,OmObservationCollection-method (OmObservationCollection), 36
sosResult,OmObservationProperty-method (OmObservation-class), 34
sosResult,OwOM_Observation-method (OwOM_Observation-class), 37
sosResult,OwsExceptionReport-method (OWS), 39
sosResult,WmlMeasurementTimeseries-method (WML), 84
sosResult,WmlMeasurementTVP-method (WML), 84
sosResultModelName (Constants), 6
sosResultModels (SOS), 56
sosResultModels,OwsOperation-method (SOS), 56
sosResultModels,SosObservationOffering-method (SosObservationOffering-class), 75
sosResultModels,SosObservationOffering_2.0.0-method (SosObservationOffering-class), 75
sosResultModels-methods (SOS), 56
sosResultName (Constants), 6
sosStartTimeName (Constants), 6
sosService (Constants), 6
sosServiceIdentification (SOS), 56
sosServiceIdentification,SOS-method (SOS), 56
sosServiceIdentification-methods (SOS), 56
sosServiceProvider (SOS), 56
sosServiceProvider,SosOperation-method (SOS), 56
sosSrsName (SOS), 56
sosSrsName,OmOM_Observation-method (OmOM_Observation-class), 37
sosSrsName,OmPoint-method (GML), 23
sosSrsName,OmPoint-method (GML), 23
sosSrsName,OmTime-method (SOS), 56
sosSrsName-methods (SOS), 56
SosSupported(Supported), 78
SosSupportedBindings, 57, 61
SosSupportedBindings (Supported), 78
SosSupportedComparisonOperators (Supported), 78
SosSupportedDataAvailabilityMember-method (SosGetDataAvailability_1.0.0-class), 73
SosSupportedGeometryOperands (Supported), 78
SosSupportedOperations (Supported), 78
SosSupportedResponseFormats (Supported), 78
SosSupportedResponseModes (Supported), 78
SosSupportedResultModels (Supported), 78
SosSupportedServiceVersions (Supported), 78
SosSupportedSpatialOperators (Supported), 78
SosSupportedTemporalOperators (Supported), 78
socSwitchCoordinates (SOS), 56
socSwitchCoordinates,SOS-method (SOS), 56
socSwitchCoordinates-method (SOS), 56
socTime (SOS), 56
socTime,DataAvailabilityMember-method (SosGetDataAvailability_1.0.0-class), 73
socTime,GmlTimeInstant-method (GML), 23
socTime,GmlTimeInstantProperty-method (GML), 23
socTime,GmlTimePeriod-method (GML), 23
socTime,GmlTimePosition-method (GML), 23
socTime,list-method (SOS), 56
socTime,OwOM_Observation-method (OwOM_Observation-class), 37
socTime,SensorML-method (SML), 54
socTime,SOS-method (SOS), 56
socTime,SosObservationOffering-method (SosObservationOffering-class), 75
socTime,SosObservationOffering_2.0.0-method (SosObservationOffering-class), 75
socTime,SosObservationOffering-method (SosObservationOffering-class), 75
socTime,WmlMeasurementTVP-method (WML), 84
socTime-methods (SOS), 56
socTimeFormat (SOS), 56
socTimeFormat,SOS-method (SOS), 56
socTimeFormat-methods (SOS), 56
socTimeName (Constants), 6
socTitle (SOS), 56
socTitle,OwsServiceIdentification-method (OWS), 39
sosTitle, SOS-method (SOS), 56
sosTitle-method (SOS), 56
sosUOM (SOS), 56
sosUOM, data.frame-method (SOS), 56
sosUOM, GmlMeasure-method (GML), 23
sosUOM, list-method (SOS), 56
sosUOM, OmMeasurement-method (OmMeasurement), 32
sosUOM, OmObservation-method (OmObservation-class), 34
sosUOM, OmObservationCollection-method (OmObservationCollection), 36
sosUOM, WmlDefaultTVPMeasurementMetadata-method (WML), 84
sosUrl (SOS), 56
sosUrl, SOS-method (SOS), 56
sosUrl, SOS_1.0.0-method (SOS), 56
sosUrl, SOS_2.0.0-method (SOS), 56
sosUrl-methods (SOS), 56
sosVersion (SOS), 56
sosVersion, SOS-method (SOS), 56
sosVersion-methods (SOS), 56
Spatial, 17
SpatialPointsDataFrame, 53
summary, GmlTimePeriod (GML), 23
summary, OmObservation (OmObservation-class), 34
summary, OmObservationCollection (OmObservationCollection), 36
summary, OwsRange (OWS), 39
summary, SOS (SOS), 56
summary, SOS_versioned (SOS), 56
summary, SosObservationOffering (SOS), 56
summary, SosObservationOffering_2.0.0 (SosObservationOffering-class), 75
Supported, 78
SWE, 79
swe20DataArrayName (Constants), 6
swe20Namespace (Constants), 6
swe20NamespacePrefix (Constants), 6
sweBaseName (Constants), 6
sweBooleanName (Constants), 6
sweCategoryName (Constants), 6
sweCodeSpaceName (Constants), 6
sweComponentName (Constants), 6
SweCompositePhenomenon (SWE), 79
SweCompositePhenomenon-class (SWE), 79
sweCompositePhenomenonName (Constants), 6
sweCoordinateName (Constants), 6
sweCountName (Constants), 6
sweDataArrayName (Constants), 6
sweDataRecordName (Constants), 6
sweDataTypeName (Constants), 6
sweEncodingName (Constants), 6
sweFieldName (Constants), 6
sweLocationName (Constants), 6
sweLowerCornerName (Constants), 6
sweNamespace (Constants), 6
sweNamespacePrefix (Constants), 6
SwePhenomenon, 81
SwePhenomenon (SWE), 79
SwePhenomenon-class (SWE), 79
SwePhenomenonOrNULL, 81
SwePhenomenonOrNULL-class (SWE), 79
SwePhenomenonProperty (SWE), 79
SwePhenomenonProperty-class (SWE), 79
SwePhenomenonPropertyOrNULL, 81
SwePhenomenonPropertyOrNULL-class (SWE), 79
swePositionName (Constants), 6
sweQuantityName (Constants), 6
sweDescribeSensorName (Constants), 6
sweDescribeSensorResponseName (Constants), 6
sweDescriptionName (Constants), 6
sweIdentifierName (Constants), 6
sweSimpleDataRecordName (Constants), 6
sweNameName (Constants), 6
sweNamespace (Constants), 6
sweNamespacePrefix (Constants), 6
sweObservablePropertyName (Constants), 6
sweOfferingName (Constants), 6
sweProcedureDescriptionFormatName (Constants), 6
sweProcedureName (Constants), 6
sweSensorDescriptionName (Constants), 6
sweValidTimeName (Constants), 6
SweTextBlock (SWE), 79
SweTextBlock-class (SWE), 79
sweTextBlockName (Constants), 6
SweTextEncoding (SweTextEncoding-class), 82
SweTextEncoding-class, 82
toString, SosFeatureOfInterest-method  
(SosFeatureOfInterest-class), 70

toString, SosFilter_Capabilities-method  
(SosFilter_Capabilities-class), 71

toString, SosGetDataAvailability_1.0.0-method  
(SosGetDataAvailability_1.0.0-class), 73

toString, SosGetObservation-method  
(GetObservation), 19

toString, SosGetObservationById-method  
(GetObservation), 19

toString, SosObservationOffering-method  
(SosObservationOffering-class), 75

toString, SosObservationOffering_2.0.0-method  
(SosObservationOffering-class), 75

toString, SweCompositePhenomenon-method  
(SWE), 79

toString, SwePhenomenon-method (SWE), 79

toString, SwePhenomenonProperty-method  
(SWE), 79

toString, SweTextBlock-method (SWE), 79

toString, SweTextEncoding-method (SWE), 79

toString, TM_After-method  
(TM_Operators), 82

toString, TM_Before-method  
(TM_Operators), 82

toString, TM_During-method  
(TM_Operators), 82

toString, TM_Equals-method  
(TM_Operators), 82

toString, WmlMonitoringPoint-method  
(MonitoringPoint-class), 29

WML, 84

WmlDefaultTVPMeasurementMetadata (WML), 84
WmlDefaultTVPMeasurementMetadata-class (WML), 84
WmlInterpolationType (WML), 84
WmlInterpolationType-class (WML), 84
WmlMeasurementTimeseries (WML), 84
WmlMeasurementTimeseries-class (WML), 84

WmlMeasurementTimeseriesMetadata (WML), 84
WmlMeasurementTimeseriesMetadata-class (WML), 84
wmlMeasurementTimeseriesName (WML), 84
WmlMeasurementTVP (WML), 84
WmlMeasurementTVP-class (WML), 84
WmlMonitoringPoint (MonitoringPoint-class), 29
WmlMonitoringPoint-class (MonitoringPoint-class), 29
wmlMonitoringPointName (Constants), 6
wmlNamespace (Constants), 6
wmlNamespacePrefix (Constants), 6
WmlTimeseries (WML), 85
WmlTimeseries-class (WML), 84
wmlTimeZoneName (Constants), 6
wmlVerticalDatumName (Constants), 6
xlinkNamespace (Constants), 6
xmlTextNodeName (Constants), 6
xsiNamespace (Constants), 6
xsiNamespacePrefix (Constants), 6
xsiTypeName (Constants), 6