Package ‘sos4R’

February 20, 2015

Type Package
Title An R client for the OGC Sensor Observation Service
Version 0.2-11
Date 2013-05-14
Depends R (>= 2.15.0), XML, RCurl (>= 1.4-1), sp, methods, graphics
Suggests spacetime (>= 0.2-0), gstat, maps, maptools, mapdata, cshapes, xts, xtable, rgdal
Description sos4R is a client for Sensor Observation Services (SOS) as specified by the Open Geospatial Consortium (OGC). It allows users to retrieve metadata from SOS web services and to interactively create requests for near real-time observation data based on the available sensors, phenomena, observations et cetera using thematic, temporal and spatial filtering.
License GPL-2
URL http://www.nordholmen.net/sos4r,
http://www.52north.org/geostatistics
Encoding UTF-8
LazyLoad TRUE
ByteCompile TRUE
Author Daniel Nuest [cre, aut]
Maintainer Daniel Nuest <daniel.nuest@uni-muenster.de>
R topics documented:

sos4R-package .......................................................... 3
checkRequest-methods ................................................ 5
Constants .............................................................. 6
Defaults ............................................................... 6
DescribeSensor ........................................................ 9
describeSensor-methods ............................................. 11
encodeKVP-methods .................................................. 12
encodeRequestKVP-methods ......................................... 12
encodeRequestSOAP-methods ....................................... 13
encodeRequestXML-methods ....................................... 13
encodeXML-methods ................................................ 14
getCapabilities-methods .......................................... 15
GetObservation ........................................................ 16
getObservation-methods ............................................ 19
GmlDirectPosition-class ........................................... 20
KML ................................................................. 25
OGC ................................................................. 26
OmMeasurement ...................................................... 28
OmObservation-class .............................................. 30
OmObservationCollection ......................................... 33
OWS ................................................................. 34
parse ............................................................... 39
read.sos ............................................................ 42
SA ................................................................. 43
SML ................................................................. 45
SOS ................................................................. 46
SosBindings ........................................................ 50
SosCapabilities_1.0.0-class ..................................... 51
SosContents-class .................................................. 53
sosConvertString .................................................. 54
sosCreate ........................................................ 55
SosEventTime ........................................................ 58
SosFeatureOfInterest-class ....................................... 60
SosFilter_Capabilities-class .................................... 61
SosObservationOffering-class ................................... 62
sosRequest-methods ................................................. 64
Supported .......................................................... 65
SWE ............................................................... 66
TM_Operators ....................................................... 68

Index 71
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

<table>
<thead>
<tr>
<th>Package:</th>
<th>sos4R</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>0.2-11</td>
</tr>
<tr>
<td>Date:</td>
<td>2013-05-14</td>
</tr>
<tr>
<td>License:</td>
<td>GPL-2</td>
</tr>
<tr>
<td>LazyLoad:</td>
<td>yes</td>
</tr>
<tr>
<td>ByteCompile:</td>
<td>yes</td>
</tr>
<tr>
<td>Depends:</td>
<td>XML, RCurl, sp</td>
</tr>
</tbody>
</table>

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 52North forum for the geostatistics community (http://geostatistics.forum.52north.org/), or consider subscribing to the mailing list (http://list.52north.org/mailman/listinfo).

The most extensive documentation is contained in the package vignette.

Author(s)

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

References

See Also

See also the package vignette, vignette(sos4R).

Examples

```r
## Not run:

# Take a SOS from the example list
sos.url = SosExampleServices()[1]

# Open the connection
sos = SOS(url = sos)

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

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

# 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)
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, method = "GET", dataFieldConverters = myConverters)
sosDataFieldConverters(mySos)
```
Methods for Function checkRequest

Description

A function to check request prior to sending them to a service. This function is automatically called during the request process and can be used to check the request for consistency with itself as well as with available metadata, and also perform additional validity checks that might not be possible with class validation.

Methods

signature(service = "SOS", operation = "DescribeSensor", verbose = "logical")  Checking a DescribeSensor request that is send to a SOS. This method currently checks the following elements:
  • operation@service attribute must be SOS
  • operation@request attribute must be DescribeSensor
  • operation@procedure must be listed in the given service’s capabilities
  • operation@outputFormat must be supported by the operations capabilities description
  • operation@method must be supported by the package implemenations. See SosBindings.

signature(service = "SOS", operation = "SosGetObservationById", verbose = "logical")  Checking a GetObservationById request. Warning: Function not implemented yet.

signature(service = "SOS", operation = "SosGetObservation", verbose = "logical")  Checking a GetObservation request. Warning: Function not implemented yet.

signature(service = "SOS", operation = "OwsGetCapabilities_1.1.0", verbose = "logical")  Checking a GetCapabilities request. Warning: Function not implemented yet.

signature(service = "SOS", operation = "OwsGetCapabilities_2.0.0", verbose = "logical")  Checking a GetCapabilities request. Warning: Function not implemented yet.
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.

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, ogcGeometryOperatorLineStringName, ogcTempOpTMEqualsName.

The OwsextensionsData() 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.

See also Defaults for default parameter settings.

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

Examples

```r
# example constants
sosNsapcePrefix
gmlNameName
sweUomName

# Data frame holding OWS exception code information
OwsExceptionsData()
```
Defaults

Usage

SosDefaultConnectionMethod()

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

SosDisabledParsers()
SosExampleServices()
SosDefaults()
SosResetParsingFunctions(sos)

Arguments

... Named references to functions to be used for the respective element during parsing, encoding order 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
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**

```r
# simple default values
show(sosDefaultCharacterEncoding)
show(sosDefaultDescribeSensorOutputFormat)
show(sosDefaultGetCapAcceptFormats)
show(sosDefaultGetCapOwsVersion)
show(sosDefaultGetCapSections)
show(sosDefaultGetOwsResponseFormat)
show(sosDefaultTimeFormat)
show(sosDefaultFilenameTimeFormat)
show(sosDefaultTempOpPropertyName)
show(sosDefaultTemporalOperator)
show(sosDefaultSpatialOpPropertyName)
show(sosDefaultTimeFormat)
show(sosDefaultConnectionMethod)

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

## End(Not run)

# functions to disable all parsing
```
DescribeSensor

SosDisabledParsers()

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

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

## Not run:
# Replace an encoding function
myEncoding <- function(object, v) {
  return(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, method = "GET", dataFieldConverters = myConverters)
sosDataFieldConverters(mySos)

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

## End(Not run)

# a list of example services
SosExampleServices()

# a named list of all defaults
SosDefaults()

# replace the parsing functions with the default ones
## Not run:
sos <- SosResetParsingFunctions(sos)

## End(Not run)
DescribeSensor

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

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

Usage

`sosDescribeSensor(service, version, procedure, outputFormat)`

Arguments

- `service`: The service attribute of the request, e.g. `"SOS"`.
- `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""`.

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(...)`. They contain the procedure identifier that is to be described by a service.

Slots

- `procedure`: Object of class "character", the identifier of the procedure.
- `outputFormat`: Object of class "character", the requested output format.
- `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

- `checkRequest` signature(service = "SOS", operation = "DescribeSensor", verbose = "logical"): Checks the contents of the request before sending it.
- `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**

```
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
encodeRequestXML(describeSensorRequest)
```

**Description**

This method sends a DescribeSensor request for a description of the given procedure to the given Sensor Observation Service instance.

**Methods**

signature(sos = "SOS", procedure = "character"): Method requests a description of the given procedure from the given SOS.
**encodeKVP-methods**  
*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 = "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](http://en.wikipedia.org/wiki/Key_Value_Pair)

---

**encodeRequestKVP-methods**  
*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.
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

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.

References

See Also
SosBindings, encodeXML
# encodeXML-methods

## Encode Classes as XML

### 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 = "SosEventTimeLatest", 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 = "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 requests 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.
GetObservation

GetObservation and GetObservationById Request Objects

Description

Classes (and their construction functions) to request observations from a Sensor Observation Service.

Usage

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

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

Value  
An object of class SosGetObservation-class or SosGetObservationById-class respectively.

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

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.

offering: Object of class "character", specifies the offering URI 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.

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

**Extends**

- **SosGetObservation**: Class "OwsServiceOperation", directly.
- **SosGetObservationById**: Class "OwsServiceOperation", directly.

**Methods**

- **checkRequest** signature(service = "SOS", operation = "SosGetObservationById", verbose = "logical") or signature(service = "SOS", operation = "GetObservation", verbose = "logical"): Check the request for validity and for compliance with the metadata available in from the given SOS.

- **encodeRequestKVP** signature(obj = "SosGetObservationById") or signature(obj = "GetObservation"): Transform the information to key-value pair format, see `encodeKVP`.

- **encodeRequestXML** signature(obj = "SosGetObservationById") or signature(obj = "GetObservation"): Transform the information to XML format, see `encodeXML`.

- **encodeRequestSOAP** signature(obj = "SosGetObservation"): Transform the information to XML format for SOAP requests.

**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

showClass("SosGetObservation")
showClass("SosGetObservationById")

observationRequest <- SosGetObservation(service = "SOS", version = "1.0.0",
offering = "temperatures",
observedProperty = list("urn:property:AirTemperature"),
responseFormat = "text/xml; subtype=&quot;om/1.0.0&quot;")
print(observationRequest)

observationByIdRequest <- SosGetObservationById(service = "SOS",
version = "1.0.0", observationId = "o_12345",
responseFormat = "text/xml; subtype=&quot;om/1.0.0&quot;")
print(observeByRequest)

## Not run:
sos <- SOS("http://mysos.net/sos")
encodeXML(observeByRequest, sos = sos)

## End(Not run)
Examples

```r
## Not run:
# by identifier
sos <- SOS(...)
os.offerings <- sosOfferings(sos)
obsId <- getObservationById(sos = sos, observationId = "o_3508493")

# TODO

## End(Not run)
```

### Description

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

### Usage

```r
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.
radius 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.
end: Object of class "GmlTimeInstantPropertyOrNULL": An object of class GmlTimeInstantProperty-class, the end time of a GmlTimePeriod-class.
endPosition: Object of class "GmlTimePositionOrNULL": An object of class GmlTimePosition-class, the end time of a GmlTimePeriod-class.
exterior: Object of class "ANY" for future use in GmlPolygon.
factor: Object of class "integer": Factor parameter for a GmlTimeInterval-class. The resolution of the time interval is to one radix ^(-factor) of the specified time unit.
feature: Object of class "GmlFeatureOrNULL": The directly contained feature in a GmlFeature-class.
featureMembers: Object of class "list": A list of GmlFeature-class in a GmlFeatureCollection-class.
frame: Object of class "character": Provides a URI reference that identifies a description of the reference system.
href: Object of class "character": Reference to a property.
id: Object of class "character": The identifier/id attribute (gml:id).
indeterminatePosition: Object of class "character": Inexact temporal positions may be expressed using the optional indeterminatePosition parameter. This takes one of the following values: after, before, now, unknown.
interior: Object of class "ANY" for future use in GmlPolygon.
interval: Object of class "character": Interval parameter for a GmlTimeInterval-class.
lowerCorner: Object of class "GmlDirectPosition": Object of class GmlDirectPosition-class, the lower (left) corner of an GmlEnvelope-class.
point: Object of class "ANY": An object of class GmlPoint in a GmlPointProperty-class.
points: Object of class "list" A list of positions for future use in GmlLineString.
posList: Object of class "ANY": A list of positions for future use in GmlLineString.
pos: Object of class "character": Character string containing the coordinates in a GmlDirectPosition-class, or a GmlDirectPosition-class in a GmlPoint-class.
posList: Object of class "list": A list of positions for future use in GmlLineString.
radix: Object of class "integer": Radix parameter for a GmlTimeInterval-class. The resolution of the time interval is to one radix ^(-factor) of the specified time unit.
relatedTimes: Object of class "list": List of related times.
srsDimension: Object of class "integer": Dimensions of the coordinate reference system as stated in the coordinate reference system definition., e.g. ‘2’.
srsName: Object of class "character": Name of the spatial reference system for bounding box, e.g. "urn:ogc:def:crs:EPSG:4326".
timeInterval: Object of class "GmlTimeIntervalOrNULL": Time interval element in a GmlTimePeriod-class, an object of class GmlTimeInterval-class.
time: Object of class "GmlTimeInstantOrNULL": The actual time as an object of class POSIXt in a GmlTimePosition-class, or an object of class GmlTimeInstant-class in a GmlTimeInstantProperty-class.
timePosition: Object of class "GmlTimePosition": An object of class GmlTimePosition-class in a GmlTimeInstant-class.
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 representation, and XML “encoding”.

Author(s)

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

References


Examples

```r
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)

# 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@time+3600))

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

---

**KML**

Methods for the Namespace *kml*

**Description**

Methods for handling responses in Keyhole Markup Language (KML) format.
At the current stage, the parsers simply returns an object from the package XML with the KML document.

**Author(s)**

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

**References**

http://www.opengeospatial.org/standards/kml

http://code.google.com/intl/de-DE/apis/kml/documentation/

**Examples**

```#```

---

**OGC**

*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.

- `propertyName`: The value for the `propertyName` attribute.
- `geometryName`: The geometry to be used in a spatial filter.
- `envelope`: The geometry to be used in a spatial filter.

**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**

- **OgcBBOX, OgcBinarySpatialOp**: Class "OgcSpatialOps", directly. Class "OgcSpatialOpsOrNULL", by class "OgcSpatialOps", distance 2.
- **OgcBinaryTemporalOp**: Class "OgcBinaryTemporalOpOrNULL", directly.
- **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>

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")

# TBD examples for construction functions

---

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 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 OmMeasurement(...).

Slots

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.
metadata: Object of class "ANY": Observation metadata.
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.
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.
procedure: Object of class "ANY": The description of a process used to generate the result. It must be suitable for the observed property.
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.
resultQuality: Object of class "ANY": Event specific quality of a result.
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.

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.

Extends

OmMeasurement: Class "OmObservation", directly. Class "OmObservationOrNULL", by class "OmObservation", distance 2.

Methods

sosResult signature(obj = "OmMeasurement"): Get the data in the measurement as a data.frame.
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

showClass("OmMeasurement")

# TBD examples for construction function

# TBD examples for sosResult
Usage

```
OmObservation(samplingTime, procedure, observedProperty, featureOfInterest, result, metadata = NA, resultTime = NULL, resultQuality = NA, parameter = NA)
OmObservationProperty(href = as.character(NA), obs = NULL)
```

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.

Author(s)

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

References


See Also

See Also as OmMeasurement-class.

Examples

showClass("OmObservation")
showClass("OmObservationProperty")
showClass("OmObservationOrNULL")

# TBD examples for construction methods
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
result <- sosResult(observation)
resultList <- sosResult(observationsList)

## 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

```r
showClass("OmObservationCollection")
```

---

**Description**

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

**Usage**

```r
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 = xmlNode(NA))
OwsRange(minimumValue = as.character(NA), maximumValue = as.character(NA),
    rangeClosure = as.character(NA), spacing = as.character(NA))
OwsServiceProvider(providerName, providersite = as.character(NA),
    serviceContact = xmlNode(NA))
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 meta-
  data 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
  response, 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 capabilities 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 document.

exceptionCode: Object of class "character", the code attribute of an OWS Exception, see OwsExceptionsData. exceptionText: 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.
checkRequest signature(service = "SOS", operation = "OwsGetCapabilities_1.1.0", verbose = "logical") or signature(service = "SOS", operation = "OwsGetCapabilities_2.0.0", verbose = "logical"): Check the given operation for validity and for compliance with the metadata of the given SOS.
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")
```

# TBD examples for construction functions

---

**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**

```
parseFile(sos, file, verbose = FALSE, ...)
parseCSV(obj, verbose = FALSE)
parseNoParsing(obj)
parseCategoryObservation(obj, sos, verbose = FALSE)
```
Arguments

obj The object to decode, normally objects of either XMLNode or XMLInternalDocument.

sos An object of class SOS-class, which may be utilized/required by some parsing functions to access other parsing functions or encoding information.
verbose A boolean value indication whether information is printed out to the console during the process - potentially a lot of output!

format A character string defining the time format to be used in strptime.

values The values to be parsed in parseValues(...).

fields Field information in parseValues(...), a named list.

file Name of the file to be parsed in sosParse(...).

encoding Encoding information in parseValues(...), an object of class SweTextBlock.

... Additional arguments that are parsed to xmlParse(...) in sosParse(...).

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!

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.

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

# parsing a XML string to an exception report object
er.doc <- xmlParseDoc(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/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)
print(er.parsed)
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]])

## End(Not run)

---

**read.sos**

*Read Data from a SOS Connection*

**Description**

This function provides a high level interface to read data from a SOS connection in an abstract and uncomplicated manner. It wraps the actual request operation (i.e. GetObservation) and also uses different terms for some properties.

**WARNING**: FUNCTION IS NOT IMPLEMENTED YET! But any ideas or input is highly appreciated.

**Usage**

```r
read.sos(sos, sensors = NA_character_, phenomena = NA_character_,
bbox = NA_character_, times = NA_character_, mergeResult = FALSE,
addLocation = FALSE, verbose = FALSE)
```

**Arguments**

- **sos**
  - The SOS to read data from.
- **sensors**
  - A character array with sensor identifier(s) to limit data to certain sensors.
- **phenomena**
  - A character array with phenomenon identifier(s) to limit data to certain phenomena.
bbox  A bounding box representation to request a certain spatial extend of data.
times  A vector of times (e.g. start and end time) as POSIXt classes or character string that can be parsed to POSIXt classes depicting a temporal extend.
mergeResult  A flag to turn merging of the results into a single data structure on or off.
addLocation  A flag to turn the adding of the location to every data set in the result on or off.
verbose  Turn verbose logging on/off.

Value
An object or a list of object from class OmObservation or OmMeasurement.

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

See Also
See Also as getObservation.

Examples
# TBD

SA  Classes of the Namespace sa

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").
- `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"): ...

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"))
```
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

xml: Object of class "XMLInternalDocument", holds the XML representation of the sensor description.

coords: Object of class "data.frame", holds the position of the sensor.

id: Object of class "character", the main identifier of the sensor.

group: Object of class "character", a naming identifier of the sensor.

description: Object of class "character", a normal text description of the sensor.

boundedBy: Object of class "matrix", the bounding box of the sensor.

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>

References


Examples

showClass("SensorML")

## Not run:
weathersos <- SOS("http://v-swe.uni-muenster.de:8080/WeatherSOS/sos")
proc1 <- sosProcedures(weathersos)[1][1]
proc1.descr <- describeSensor(weathersos, proc1, verbose = TRUE)
plot(proc1.descr)
class(proc1.descr)
print(proc1.descr)

## End(Not run)

---

**SOS**

*Class, and Construction and Accessor Functions for “SOS”*

Description

Base class of a connection to a Sensor Observation Service.

Usage

```r
SOS(url, method = SosDefaultConnectionMethod(), version = "1.0.0",
parsers = SosParsingFunctions(), encoders = SosEncodingFunctions(),
dataFieldConverters = SosDataFieldConvertingFunctions(),
curlOptions = list(), curlHandle = getCurlHandle(),
timeFormat = sosDefaultTimeFormat, verboseOutput = FALSE,
switchCoordinates = FALSE, ...)
```

# accessor methods ...

Arguments

- `url` See the corresponding slot description.
- `method` See the corresponding slot description.
- `version` See the corresponding slot description.
- `parsers` See the corresponding slot description.
- `encoders` See the corresponding slot description.
- `dataFieldConverters` See the corresponding slot description.
- `curlOptions` See the corresponding slot description.
- `curlHandle` See the corresponding slot description.
- `timeFormat` See the corresponding slot description.
- `verboseOutput` See the corresponding slot description.
SOS

`switchCoordinates`

See the corresponding slot description.

... Additional parameters that are passed on to the `getObservation` call that is done within this function.

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`.
- `method`: Object of class "character": The method, or transport protocol, see `SosSupportedConnectionMethods` 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" ~~
- `encoders`: Object of class "list" ~~
- `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.
- `curlHandle`: Object of class "CURLHandle": The curl options to be passed on to the methods `getURL` and `postForm` from the package `RCurl`.
- `curlOptions`: Object of class "ANY": The curl handle to be passed on to the methods `getURL` and `postForm` from the package `RCurl`.
- `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 switching of lat/long during parsing stage.
Methods

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

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"): ...

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.

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

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.

sosFilterCapabilities signature(sos = "SOS"): Accessor function for the filter capabilities of a 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 http://www.opengeospatial.org/standards/sos.

See Also

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

Examples

showClass("SOS")

## Not run:
# create a SOS connection
mysos <- SOS(url = "http://mysos.org/sos")

# create a SOS connection with a specific connection method and time format
mysos <- SOS(url = "http://mysos.org/sos",
method = "GET", timeFormat = "

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

## 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.

The GET binding is described by OOTethys in a Best Practice document: http://www.oostethys.org/best-practices/best-practices-get. 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

SosSupportedConnectionMethods

Examples

# HTTP connection methods supported by this sos4R implementation
supported <- SosSupportedConnectionMethods()
supported

## Not run:
sos <- SOS("http://sosurl.org/", method = "GET")

## End(Not run)
Description

The service metadata document of a Sensor Observation Service.

Usage

SosCapabilities(version, updateSequence = NA, owsVersion = "1.1.0",
identification = NULL, provider = NULL, operations = NULL,
fILTERCapabilities = NULL, contents = NULL)

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 a OperationsMetadata object. The
provider section of a capabilities document.

FILTERCapabilities An object of class SosFilterCapabilities 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 spec-
ification 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.
SosCapabilities_1.0.0-class

Slots

- `filterCapabilities`: Object of class "SosFilterCapabilitiesOrNULL" ~
- `identification`: Object of class "OwsServiceIdentificationOrNULL" ~
- `provider`: Object of class "OwsServiceProviderOrNULL" ~
- `operations`: Object of class "OwsOperationsMetadataOrNULL" ~
- `contents`: Object of class "OwsContentsOrMetadataOrNULL" ~
- `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

SosFilterCapabilities, SosContents, OwsCapabilities

Examples

showClass("SosCapabilities_1.0.0")
**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.

**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.
SosConversionString

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>
sosCreate

Convenience Functions for Request Parameter Creations

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.

Usage

sosCreateBBox(lowLat, lowLon, uppLat, uppLon, srsName,
               srsDimension = NA_integer_, axisLabels = NA_character_,
               uomLabels = NA_character_, propertyName = sosDefaultSpatialOpPropertyName)
sosCreateFeatureOfInterest(objectIDs = list(NA), spatialOps = NULL, bbox = NULL,
                           srsName = NA_character_)
sosCreateEventTime(time, operator)
sosCreateEventTimeList(time, operator)
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),

Examples

```r
## Not run:
sos <- SOS(url = SosExampleServices()[2])
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)
```
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: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’.
- **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.
- **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.
- **end** Object of class POSIXt.
- **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: GmlTimeInstant-class, GmlTimePeriod-class, SosEventTime-class, SosFeatureOfInterest-class, OgcBBOX-class, matrix-class.

Examples

# 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 <- sosCreateBBBoxMatrix(lowLat = 50.0, lowLon = 7.0,
upplat = 53.0, upplon = 10.0)
foiBBBox <- sosCreateFeatureOfInterest(bbox = bboxMatrix,
srsName = "urn:ogc:def:crs:EPSG:6.8:4326")
print(foiBBBox)

# 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")
foiBBBox2 <- sosCreateFeatureOfInterest(spatialOps = bbox)
print(foiBBBox2)

## Not run:
lastPeriod <- sosCreateTimePeriod(sos = mySOS,
begin = Sys.time() - 3600 * 24 * 7), end = Sys.time())

oneWeekPeriod <- sosCreateTimePeriod(sos = mySOS,
begin = as.POSIXct("2010/01/01"), end = as.POSIXct("2010/01/07"))
oneWeek.eventTime <- sosCreateEventTimeList(oneWeekPeriod)
sosCreateTime(sos = mySOS, time = "2007-07-07 07:00::2008-08-08 08:00")
soscreateTime(sos = mySOS, time = "2007-07-07 07:00/2010-10-10 10:00")
soscreateTime(sos = mySOS, time = ":2007-08-05")
Classes and Construction Functions for sos:eventTime elements.

Description
Temporal query parameters for GetObservation requests.

Usage
SosEventTime(temporalOps)
SosEventTimeLatest()

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 (paramter in GetObservation).

The request for the latest observation is not standard conform. The distinction between the classes happens eventually during encoding, where both classes have different encoding functions.

A typical example in a POST request:
<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(...) or SosEventTimeLatest(...).
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

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

encodeXML signature(obj = "SosEventTime", sos = "SOS") or signature(obj = "SosEventTimeLatest", 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")
showClass("SosEventTimeLatest")

## Not run:
# create SosEventTime for all times after the given time stamp
tOps <- TM_After(time = GmlTimeInstant(timePosition = GmlTimePosition(as.POSIXct("2010-01-01 12:00")))))
time1 <- SosEventTime(tOps)

# encode it as XML and KVP
encodeXML(time1)
encodeKVP(time1)

time2 <- SosEventTimeLatest()
encodeXML(time2)

## End(Not run)
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 0gcSpatialOps 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 it.

Slots

- objectIDs: Object of class "list": Identifiers of features of interest.
- spatialOps: Object of class "0gcSpatialOpsOrNULL": A spatial filtering of the result.

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
spatial A character list of names of available spatial filters.
temporal A character list of names of available temporal filters.
scalar A character list of names of available scalar filters.
id A character list of names of available filters on identifiers.

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", ...). SosFilter_CapabilitiesOrNULL is virtual class: No objects may be created from it.
SosObservationOffering-class

Slots
- **spatial**: Object of class "list" with character strings for names of spatial filters.
- **temporal**: Object of class "list" with character strings for names of temporal filters.
- **scalar**: Object of class "list" with character strings for names of scalar filters.
- **id**: Object of class "list" with character strings for names of ID filters.

Extends
Class **"SosFilter_CapabilitiesOrNULL"**, directly.

Methods
- **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
- **SosCapabilities**

Examples
```
showClass("SosFilter_Capabilities")
showClass("SosFilter_CapabilitiesOrNULL")
```

Description
SosObservationOfferings collect all metadata about a specific offerign in a Sensor Observation Service.

Usage
```
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")
# TBD examples for construction functions

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.
Supported

Methods

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

SosSupportedOperations()
SosSupportedComparisonOperators()
SosSupportedConnectionMethods()
SosSupportedGeometryOperands()
SosSupportedResponseFormats()
SosSupportedResponseModes()
SosSupportedResultModels()
SosSupportedSpatialOperators()
SosSupportedTemporalOperators()
SosSupportedServiceVersions()

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 func-
tions.

It is encouraged to rather use these methods than manually set character values for compatibility
with future versions, e.g. SosSupportedConnectionMethods()[[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
SosSupportedConnectionMethods()
myConnectionType <- SosSupportedConnectionMethods()[[1]]
myConnectionType

# 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()

---

SWE

Classes and Construction Functions for the SWE Namespace

Description

These classes represent elements from the OpenGIS(R) Sensor Model Language (SensorML) Implementation Specification that are used to model observation data in responses from a Sensor Observation Service.

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 a inline property.

phenomenon: The inline phenomenon of a phenomenon property.

tokenSeparator: The character to be used as the token seperator, often ",".

blockSeparator: The character to be used as the block seperator, often ";".

decimalSeparator: The character to be used as the decimal seperator, 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 seperator 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 seperator 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 seperator in a SweTextBlock, e.g. in the case of "." this would result in attribute1,42.0,attribute3,23.0.
Extends

SweCompositePhenomenon: Class "SwePhenomenon", directly. Class "SwePhenomenonOrNULL", by class "SwePhenomenon", distance 2.

SwePhenomenon: Class "SwePhenomenonOrNULL", directly.

SwePhenomenonProperty: Class "SwePhenomenonPropertyOrNULL", 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")

# TBD examples for construction functions

Description

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

Usage

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

codeXML signature(obj = "TM_After", sos = "SOS"): ...
show signature(object = "TM_After"): ...
codeXML signature(obj = "TM_Before", sos = "SOS"): ...
show signature(object = "TM_Before"): ...
codeXML signature(obj = "TM_During", sos = "SOS"): ...
show signature(object = "TM_During"): ...
codeXML 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)
Index

*Topic **XML**
  Constants, 6
  encodeRequestXML-methods, 13
  SosBindings, 50

*Topic **classes**
  DescribeSensor, 9
  GetObservation, 16
  GmlDirectPosition-class, 20
  OGC, 26
  OmMeasurement, 28
  OmObservation-class, 30
  OmObservationCollection, 33
  OWS, 34
  SA, 43
  SML, 45
  SOS, 46
  SosCapabilities_1.0.0-class, 51
  SosContents-class, 53
  SosEventTime, 58
  SosFeatureOfInterest-class, 60
  SosFilter_Capabilities-class, 61
  SosObservationOffering-class, 62
  SWE, 66
  TM_Operators, 68

*Topic **connection**
  sos4R-package, 3

*Topic **constants**
  Constants, 6
  SosBindings, 50

*Topic **database**
  sos4R-package, 3

*Topic **methods**
  checkRequest-methods, 5
  describeSensor-methods, 11
  encodeKVP-methods, 12
  encodeRequestKVP-methods, 12
  encodeRequestSOAP-methods, 13
  encodeRequestXML-methods, 13
  encodeXML-methods, 14
  getCapabilities-methods, 15
  getObservation-methods, 19
  KML, 25
  parse, 39
  sosCreate, 55
  sosRequest-methods, 64

*Topic **misc**
  Defaults, 6
  KML, 25
  parse, 39

*Topic **package**
  sos4R-package, 3

*Topic **spatial**
  sos4R-package, 3

*Topic **ts**
  sos4R-package, 3

*Topic **utilities**
  GmlDirectPosition-class, 20
  OGC, 26
  OWS, 34
  read.sos, 42
  sosConvertString, 54
  sosCreate, 55
  Supported, 65
  TM_Operators, 68

*Topic **utilities**
  GetObservation, 16
  [[,OmObservationCollection-method
  (OmObservationCollection), 33
  [[,OmObservationCollection,ANY,missing-method
  (OmObservationCollection), 33
  as.data.frame.OmMeasurement
  (OmMeasurement), 28
  as.data.frame.OmObservation
  (OmObservation-class), 30
  as.list.OmObservationCollection
  (OmObservationCollection), 33
  as.SensorML.SpatialPointsDataFrame
  (SML), 45
as.SosObservationOffering.SpatialPolygons
(SOS), 46
as.SpatialPointsDataFrame.OmMeasurement
(OmMeasurement), 28
as.SpatialPointsDataFrame.OmObservation
(OmObservation-class), 30
as.SpatialPointsDataFrame.OmObservationCollection
(OmObservationCollection), 33

bindings (SosBindings), 50

checkRequest (checkRequest-methods), 5
checkRequest, SOS, OwsGetCapabilities_1.1.0, logical-method
(checkRequest-methods), 5
checkRequest, SOS, OwsGetCapabilities_2.0.0, logical-method
(checkRequest-methods), 5
checkRequest, SOS, SosDescribeSensor, logical-method
(checkRequest-methods), 5
checkRequest, SOS, SosGetObservation, logical-method
(checkRequest-methods), 5
checkRequest, SOS, SosGetObservationById, logical-method
(checkRequest-methods), 5
checkRequest-methods, 5

connection methods (SosBindings), 50

Constants, 6, 8

Defaults, 6, 6, 65

DescribeSensor, 9
describeSensor, 10, 11
describeSensor
(describeSensor-methods), 11
describeSensor, SOS, character-method
(describeSensor-methods), 11
describeSensor, SOS_1.0.0, character-method
(describeSensor-methods), 11
describeSensor-methods, 11

codeKVP, 18, 27
codeKVP (codeKVP-methods), 12
codeKVP, OgcBinaryTemporalOp, SOS-method
(codeKVP-methods), 12
codeKVP, POSIXt, SOS-method
(codeKVP-methods), 12
codeKVP, SosEventTime, ANY-method
(codeKVP-methods), 12
codeKVP, SosEventTime, SOS-method
(codeKVP-methods), 12
codeKVP, SosEventTimeLatest, SOS-method
(codeKVP-methods), 12
codeKVP-methods, 12
(encodeRequestXML-methods), 13
encodeRequestXML, XMLNode, SOS-method
(encodeRequestXML-methods), 13
encodeRequestXML-methods, 13
encodeXML, 13, 18, 27
encodeXML (encodeXML-methods), 14
encodeXML, character, SOS-method
(encodeXML-methods), 14
encodeXML, GmlDirectPosition, SOS-method
(encodeXML-methods), 14
encodeXML, GmlEnvelope, SOS-method
(encodeXML-methods), 14
encodeXML, GmlEnvelope, SOS-method
(encodeXML-methods), 14
encodeXML, GmlLineString, SOS-method
(encodeXML-methods), 14
encodeXML, GmlPoint, SOS-method
(encodeXML-methods), 14
encodeXML, GmlPointProperty, SOS-method
(encodeXML-methods), 14
encodeXML, GmlPolygon, SOS-method
(encodeXML-methods), 14
encodeXML, GmlTimeInstant, SOS-method
(encodeXML-methods), 14
encodeXML, GmlTimeInstantProperty, SOS-method
(encodeXML-methods), 14
encodeXML, GmlTimePeriod, SOS-method
(encodeXML-methods), 14
encodeXML, GmlTimePosition, SOS-method
(encodeXML-methods), 14
encodeXML, OgcBBOX, SOS-method
(encodeXML-methods), 14
encodeXML, OgcComparisonOps, SOS-method
(encodeXML-methods), 14
encodeXML, OgcContains, SOS-method
(encodeXML-methods), 14
encodeXML, OgcIntersects, SOS-method
(encodeXML-methods), 14
encodeXML, OgcOverlaps, SOS-method
(encodeXML-methods), 14
encodeXML, POSIXt, SOS-method
(encodeXML-methods), 14
encodeXML, SosEventTime, SOS-method
(encodeXML-methods), 14
encodeXML, SosEventTimeLatest, SOS-method
(encodeXML-methods), 14
encodeXML, SosFeatureOfInterest, SOS-method
(encodeXML-methods), 14
encodeXML, TM_After, SOS-method
(encodeXML-methods), 14
encodeXML, TM_Before, SOS-method
(encodeXML-methods), 14
encodeXML, TM_During, SOS-method
(encodeXML-methods), 14
encodeXML, TM_Equals, SOS-method
(encodeXML-methods), 14
encodeXML, XMLInternalElementNode, SOS-method
(encodeXML-methods), 14
encodeXML, XMLNode, SOS-method
(encodeXML-methods), 14
encodeXML-methods, 14
GET (SosBindings), 50
getCapabilities
( getCapabilities-methods), 15
getCapabilities, SOS-method
( getCapabilities-methods), 15
getCapabilities, Sos1.0.0, verbose, inspect-method
( getCapabilities-methods), 15
getCapabilities, Sos1.0.0-method
( getCapabilities-methods), 15
getCapabilities-methods, 15
GetObservation, 16, 58
getObservation, 43, 47
getObservation
( getObservation-methods), 19
getObservation, SOS, character-method
( getObservation-methods), 19
getObservation, SOS, SosObservationOffering-method
( getObservation-methods), 19
getObservation, Sos1.0.0, character-method
( getObservation-methods), 19
getObservation, Sos1.0.0, SosObservationOffering-method
( getObservation-methods), 19
getObservation-methods, 19
GetObservationById (GetObservation), 16
gmlBeginName (Constants), 6
gmlBeginPositionName (Constants), 6
gmlBoundedByName (Constants), 6
gmlDescriptionName (Constants), 6
GmlDirectPosition (GmlDirectPosition-class), 20
GmlDirectPosition-class, 20
GmlDirectPositionLocation (GmlDirectPosition-class), 20
GmlDirectPositionOrNULL-class (GmlDirectPosition-class), 20
gmlDurationName (Constants), 6
gmlEndName (Constants), 6
gmlEndPositionName (Constants), 6
GmlEnvelope (GmlDirectPosition-class), 20
GmlEnvelope-class (GmlDirectPosition-class), 20
gmlEnvelopeName (Constants), 6
GmlFeature, 22, 44
GmlFeature-class (GmlDirectPosition-class), 20
GmlFeatureCollection (GmlDirectPosition-class), 20
GmlFeatureCollection-class (GmlDirectPosition-class), 20
gmlFeatureCollectionName (Constants), 6
gmlFeatureMemberName (Constants), 6
GmlFeatureOrGmlFeaturePropertyOrNULL-class (GmlDirectPosition-class), 20
GmlFeatureOrNULL, 22, 44
GmlFeatureOrNULL-class (GmlDirectPosition-class), 20
GmlFeatureProperty (GmlDirectPosition-class), 20
GmlFeatureProperty-class (GmlDirectPosition-class), 20
GmlGeometry, 22
GmlGeometry-class (GmlDirectPosition-class), 20
GmlLineString-class (GmlDirectPosition-class), 20
gmlLowerCornerName (Constants), 6
GmlMeasure, 29, 30
GmlMeasure (GmlDirectPosition-class), 20
GmlMeasure-class (GmlDirectPosition-class), 20
GmlNameName (Constants), 6
gmlNamespacePrefix (Constants), 6
GmlPoint (GmlDirectPosition-class), 20
GmlPoint-class (GmlDirectPosition-class), 20
gmlPointName (Constants), 6
GmlPointOrNULL-class (GmlDirectPosition-class), 20
GmlPointProperty (GmlDirectPosition-class), 20
GmlPointProperty-class (GmlDirectPosition-class), 20
GmlPolygon-class (GmlDirectPosition-class), 20
gmlPosName (Constants), 6
gmlRelatedTimeName (Constants), 6
GmlTimeGeometricPrimitive, 22
GmlTimeGeometricPrimitive-class (GmlDirectPosition-class), 20
GmlTimeInstant (GmlDirectPosition-class), 20
GmlTimeInstant-class (GmlDirectPosition-class), 20
gmlTimeInstantName (Constants), 6
GmlTimeInstantOrNULL, 22
GmlTimeInstantOrNULL-class (GmlDirectPosition-class), 20
GmlTimeInstantProperty (GmlDirectPosition-class), 20
GmlTimeInstantProperty-class (GmlDirectPosition-class), 20
GmlTimeInstantPropertyOrNULL, 22
GmlTimeInstantPropertyOrNULL-class (GmlDirectPosition-class), 20
GmlTimeInterval (GmlDirectPosition-class), 20
GmlTimeInterval-class (GmlDirectPosition-class), 20
gmlTimeIntervalName (Constants), 6
GmlTimeIntervalOrNULL, 22
GmlTimeIntervalOrNULL-class (GmlDirectPosition-class), 20
gmlTimeIntervalLengthName (Constants), 6
GmlTimeObject, 22
GmlTimeObject-class (GmlDirectPosition-class), 20
GmlTimeObjectOrNULL, 22
GmlTimeObjectOrNULL-class (GmlDirectPosition-class), 20
GmlTimePeriod (GmlDirectPosition-class), 20
GmlTimePeriod-class (GmlDirectPosition-class), 20
INDEX

gmLitePeriodName (Constants), 6
GmlTimePosition
  (GmlDirectPosition-class), 20
GmlTimePosition-class
  (GmlDirectPosition-class), 20
gmLitePositionName (Constants), 6
GmlTimePositionOrNULL-class
  (GmlDirectPosition-class), 20
GmlTimePrimitive, 22
GmlTimePrimitive-class
  (GmlDirectPosition-class), 20
gmlUpperCornerName (Constants), 6

HTTP (SosBindings), 50
KML, 25
kmL (KML), 25
kmlName (KML), 25

length, OmObservationCollection-method
  (OmObservationCollection), 33
length, OmObservationCollection
  (OmObservationCollection), 33

mimeSubtypeOM (Constants), 6
mimeTypeCSV (Constants), 6
mimeTypeKML (KML), 25
mimeTypeOM (Constants), 6
mimeTypeSML (Constants), 6
mimeXMLM (Constants), 6

names, OmMeasurement (OmMeasurement), 28
names, OmObservation
  (OmObservation-class), 30
names, OmObservationCollection
  (OmObservationCollection), 33

OOG, 26
ogc (OGC), 26
ogcArithmeticOperatorsName (Constants), 6
OgcBinaryOps (OGC), 26
OgcBinaryOps-class (OGC), 26
ogcBinaryOpsName (Constants), 6
OgcBinarySpatialOp, 27
OgcBinarySpatialOp-class (OGC), 26
OgcBinarySpatialOp (OGC), 26
OgcBinaryTemporalOp, 26
OgcBinaryTemporalOp-class (OGC), 26
OgcOverlaps (OGC), 26
OwsException-class (OWS), 34
owsExceptionName (Constants), 6
OwsExceptionReport (OWS), 34
OwsExceptionReport-class (OWS), 34
owsExceptionReportName (Constants), 6
OwsExceptionsData, 35, 37
OwsExceptionsData (Constants), 6
owsExceptionTextName (Constants), 6
ows FeesName (Constants), 6
OwsGetCapabilities, 38
OwsGetCapabilities (OWS), 34
OwsGetCapabilities-class (OWS), 34
OwsGetCapabilities_1.1.0, 38
OwsGetCapabilities_1.1.0-class (OWS), 34
OwsGetCapabilities_2.0.0-class (OWS), 34
owsGetName (Constants), 6
owsHTTPName (Constants), 6
owsKeywordName (Constants), 6
owsKeywordsName (Constants), 6
owsMaximumValueName (Constants), 6
owsMetadataName (Constants), 6
owsMinimumValueName (Constants), 6
owsNamespacePrefix (Constants), 6
OwsOperation (OWS), 34
OwsOperation-class (OWS), 34
owsOperationName (Constants), 6
OwsOperationsMetadata (OWS), 34
OwsOperationsMetadata-class (OWS), 34
owsOperationsMetadataName (Constants), 6
OwsOperationsMetadataOrNULL-class (OWS), 34
owsParameterName (Constants), 6
owsPostName (Constants), 6
owsProfileName (Constants), 6
owsProviderNameName (Constants), 6
owsProviderSiteName (Constants), 6
OwsRange (OWS), 34
OwsRange-class (OWS), 34
owsRangeName (Constants), 6
owsServiceContactName (Constants), 6
OwsServiceIdentification (OWS), 34
OwsServiceIdentification-class (OWS), 34
owsServiceIdentificationName (Constants), 6
OwsServiceIdentificationOrNULL, 38
OwsServiceIdentificationOrNULL-class (OWS), 34
OwsServiceOperation, 10, 18, 38
OwsServiceOperation-class (OWS), 34
OwsServiceProvider (OWS), 34
OwsServiceProvider-class (OWS), 34
owsServiceProviderName (Constants), 6
OwsServiceProviderOrNULL, 38
OwsServiceProviderOrNULL-class (OWS), 34
owsServiceProviderType (Constants), 6
owsServiceProviderTypeVersionName (Constants), 6
owsSpacingName (Constants), 6
ows TitleName (Constants), 6
owsValueName (Constants), 6
parse, 39
parseCategoryObservation (parse), 39
parseComplexObservation (parse), 39
parseComponent (parse), 39
parseCompositePhenomenon (parse), 39
parseCoordinate (parse), 39
parseCoordinate-method (parse), 39
parseCountObservation (parse), 39
parseCSV (parse), 39
parseDataArray (parse), 39
parseElementType (parse), 39
parseEncoding (parse), 39
parseFeatureCollection (parse), 39
parseFile (parse), 39
parseFile, SOS_1.0.0, character-method (parse), 39
parseFile-method (parse), 39
parseFOI (parse), 39
parseGeometryObservation (parse), 39
parseKML (KML), 25
parseLocation (parse), 39
parseLocation-method (parse), 39
parseMeasure (parse), 39
parseMeasurement (parse), 39
parseNoParsing (parse), 39
parseObservation (parse), 39
parseObservationCollection (parse), 39
parseObservationProperty (parse), 39
parseOM (parse), 39
parseOwsException (parse), 39
parseOwsExceptionReport (parse), 39
parseOwsOperation (parse), 39
parseOwsRange (parse), 39
parseOwsServiceIdentification (parse), 39
parseOwsServiceProvider (parse), 39
INDEX

parsePhenomenonProperty (parse), 39
parsePoint (parse), 39
parsePosition (parse), 39
parseResult (parse), 39
parseSamplingPoint (parse), 39
parseSamplingTime (parse), 39
parseSensorML (parse), 39
parseSosCapabilities (parse), 39
parseSosFilter_Capabilities (parse), 39
parseSosObservationOffering (parse), 39
parseSwePosition (parse), 39
parseSwePosition-method (parse), 39
parseTemporalObservation (parse), 39
parseTextBlock (parse), 39
parseTimeGeometricPrimitiveFromParent (parse), 39
parseTimeInstant (parse), 39
parseTimeInstantProperty (parse), 39
parseTimePeriod (parse), 39
parseTimePosition (parse), 39
parseTruthObservation (parse), 39
parseValues (parse), 39
parseVector (parse), 39
parseVector-method (parse), 39
plot, SensorML, missing-method (SML), 45
plot, SOS, missing-method (SOS), 46
plot, SosObservationOffering, missing-method (SOS), 46
plot, SensorML (SML), 45
plot, SOS (SOS), 46
plot, SosObservationOffering (SOS), 46
POST (SosBindings), 50
print, GmlDirectPosition-method (GmlDirectPosition-class), 20
print, GmlTimeInstant-method (GmlDirectPosition-class), 20
print, GmlTimeInstantProperty-method (GmlDirectPosition-class), 20
print, GmlTimeInterval-method (GmlDirectPosition-class), 20
print, GmlTimePeriod-method (GmlDirectPosition-class), 20
print, GmlTimePosition-method (GmlDirectPosition-class), 20
print, OgcBBOX-method (OGC), 26
print, OgcContains-method (OGC), 26
print, OgcIntersects-method (OGC), 26
print, OgcOverlaps-method (OGC), 26
print, OmMeasurement-method (OmMeasurement), 28
print, OmObservation-method (OmObservation-class), 30
print, OmObservationCollection-method (OmObservationCollection), 33
print, OmObservationProperty-method (OmObservation-class), 30
print, OwsCapabilities-method (OWS), 34
print, OwsCapabilities_1.1.0-method (OWS), 34
print, OwsCapabilities_2.0.0-method (OWS), 34
print, OwsContents-method (OWS), 34
print, OwsException-method (OWS), 34
print, OwsExceptionReport-method (OWS), 34
print, OwsGetCapabilities-method (OWS), 34
print, OwsGetCapabilities_1.1.0-method (OWS), 34
print, OwsGetCapabilities_2.0.0-method (OWS), 34
print, OwsOperation-method (OWS), 34
print, OwsOperationsMetadata-method (OWS), 34
print, OwsRange-method (OWS), 34
print, OwsServiceIdentification-method (OWS), 34
print, OwsServiceOperation-method (OWS), 34
print, OwsServiceProvider-method (OWS), 34
print, SaSamplingPoint-method (SA), 43
INDEX

print, SaSamplingSurface-method (SA), 43
print, SensorML-method (SML), 45
print, SOS-method (SOS), 46
print, SOS_1.0.0-method (SOS), 46
print, SosContents-method (SosContents-class), 53
print, SosDescribeSensor-method (DescribeSensor), 9
print, SosEventTime-method (SosEventTime), 58
print, SosFeatureOfInterest-method (SosFeatureOfInterest-class), 60
print, SosFilter_Capabilities-method (SosFilter_Capabilities-class), 61
print, SosGetObservation-method (GetObservation), 16
print, SosGetObservationById-method (GetObservation), 16
print, SosObservationOffering-method (SosObservationOffering-class), 62
print, SweCompositePhenomenon-method (SWE), 66
print, SwePhenomenon-method (SWE), 66
print, SwePhenomenonProperty-method (SWE), 66
print, SweTextBlock-method (SWE), 66
print, TM_After-method (TM_Operators), 68
print, TM_Before-method (TM_Operators), 68
print, TM_During-method (TM_Operators), 68
print, TM_Equals-method (TM_Operators), 68
print.summary.GmlTimePeriod-method (GmlDirectPosition-class), 20
print.summary.OMObservation-method (OMObservation-class), 30
print.summary.OMObservationCollection-method (OMObservationCollection), 33
print.summary.OwsRange-method (OWS), 34
print.summary.SOS-method (SOS), 46
print.summary.SosObservationOffering-method (SOS), 46
read.sos.42

Sa (SA), 43
sampling features (SA), 43
saPositionName (Constants), 6
saSampledFeatureName (Constants), 6
SaSamplingPoint (SA), 43
SaSamplingPoint-class (SA), 43
saSamplingPointName (Constants), 6
SaSamplingSurface (SA), 43
saSamplingSurface (Constants), 6
SaSamplingSurface-class (SA), 43
saSamplingStartTimeName (Constants), 6
SensorML, 11
SensorML-class (SML), 45
show, GmlDirectPosition-method (GmlDirectPosition-class), 20
show, GmlEnvelope-method (GmlDirectPosition-class), 20
show, GmlFeatureCollection-method (GmlDirectPosition-class), 20
show, GmlFeatureProperty-method (GmlDirectPosition-class), 20
show, GmlGeometry-method (GmlDirectPosition-class), 20
show, GmlMeasure-method (GmlDirectPosition-class), 20
show, GmlPoint-method (GmlDirectPosition-class), 20
show, GmlPointProperty-method (GmlDirectPosition-class), 20
show, GmlTimeInstant-method (GmlDirectPosition-class), 20
show, GmlTimeInterval-method (GmlDirectPosition-class), 20
show, GmlTimePeriod-method (GmlDirectPosition-class), 20
show, GmlTimePosition-method (GmlDirectPosition-class), 20
show, OgcBBOX-method (OGC), 26
show, OgcContains-method (OGC), 26
show, OgcIntersects-method (OGC), 26
show, OgcOverlaps-method (OGC), 26
show, OmMeasurement-method (OmMeasurement), 28
show, OmObservation-method (OmObservation-class), 30
show, OmObservationCollection-method
show, OmObservationProperty-method
(OmObservation-class), 30
show, OwsCapabilities-method (Ows), 34
show, OwsCapabilities_1.1.0-method
(Ows), 34
show, OwsCapabilities_2.0.0-method
(Ows), 34
show, OwsContents-method (Ows), 34
show, OwsException-method (Ows), 34
show, OwsExceptionReport-method (Ows), 34
show, OwsGetCapabilities-method (Ows), 34
show, OwsGetCapabilities_1.1.0-method
(Ows), 34
show, OwsGetCapabilities_2.0.0-method
(Ows), 34
show, OwsOperation-method (Ows), 34
show, OwsOperationsMetadata-method
(Ows), 34
show, OwsRange-method (Ows), 34
show, OwsServiceIdentification-method
(Ows), 34
show, OwsServiceOperation-method (Ows), 34
show, OwsServiceProvider-method (Ows), 34
show, SaSamplingPoint-method (Sa), 43
show, SaSamplingSurface-method (Sa), 43
show, SensorML-method (Sml), 45
show, Sos-method (Sos), 46
show, Sos_1.0.0-method (Sos), 46
show, SosContents-method
(SosContents-class), 53
show, SosDescribeSensor-method
(DescribeSensor), 9
show, SosEventTime-method
(SosEventTime), 58
show, SosFeatureOfInterest-method
(SosFeatureOfInterest-class), 60
show, SosFilter_Capabilities-method
(SosFilter_Capabilities-class), 61
show, SosGetObservation-method
(GetObservation), 16
show, SosGetObservationById-method
(GetObservation), 16
show, SosObservationOffering-method
(SosObservationOffering-class), 62
show, SweCompositePhenomenon-method
(SWE), 66
show, SwePhenomenon-method (SWE), 66
show, SwePhenomenonProperty-method
(SWE), 66
show, SweTextBlock-method (SWE), 66
show, TM_After-method (TM_Operators), 68
show, TM_Before-method (TM_Operators), 68
show, TM_During-method (TM_Operators), 68
show, TM_Equals-method (TM_Operators), 68
SML, 45
sml (SML), 45
smlSensorMLName (Constants), 6
SOAP (SosBinding), 50
SOS, 46, 48, 49
SOS-class, 40
SOS-class (SOS), 46
sos4R (sos4R-package), 3
sos4R-package, 3
SOS_1.0.0 (SOS), 46
SOS_1.0.0-class (SOS), 46
sosAbstract (SOS), 46
sosAbstract, OwsServiceIdentification-method
(Ows), 34
sosAbstract, SensorML-method (SML), 45
sosAbstract, SOS-method (SOS), 46
sosAbstract-method (SOS), 46
sosAttributeFileName (Constants), 6
SosBinding (SosBinding), 50
SosBindings, 5, 12, 13, 17, 50, 58
sosBoundedBy (SOS), 46
sosBoundedBy, list-method (SOS), 46
sosBoundedBy, OmObservationCollection-method
(OmObservationCollection), 33
sosBoundedBy, SensorML-method (SML), 45
sosBoundedBy, SosObservationOffering-method
(SosObservationOffering-class), 62
sosBoundedBy-method (SOS), 46
SosCapabilities, 62, 64
SosCapabilities

  (SosCapabilities_1.0.0-class), 51
SosCapabilities-class

  (SosCapabilities_1.0.0-class), 51
SosCapabilities_1.0.0-class, 51
sosCapabilitiesDocumentOriginal (SOS), 46
sosCapabilitiesDocumentOriginal, SOS-method (SOS), 46
sosCapabilitiesName (Constants), 6
sosCapabilitiesUrl (SOS), 46
sosCapabilitiesUrl, SOS-method (SOS), 46
sosCapabilitiesUrl-method (SOS), 46
sosCaps (SOS), 46
sosCaps, SOS-method (SOS), 46
sosCaps-methods (SOS), 46
sosChanges (sos4R-package), 3
sosCheatSheet (sos4R-package), 3
SosContents, 52, 64
SosContents, class (SosContents-class), 53
sosContents (SOS), 46
sosContents, SOS-method (SOS), 46
SosContents-class, 53
sosContents-methods (SOS), 46
sosContentsName (Constants), 6
SosContentsOrNULL, 53
SosContentsOrNULL-class
(sosContents-class), 53
sosConvertDouble (sosConvertString), 54
sosConvertLogical (sosConvertString), 54
sosConvertString, 54
sosConvertTime (sosConvertString), 54
sosConvertTime-method (SOS), 46
sosCoordinates, GmlDirectPosition-method (GmlDirectPosition-class), 20
sosCoordinates, GmlFeatureCollection-method (GmlDirectPosition-class), 20
sosCoordinates, GmlFeatureProperty-method (GmlDirectPosition-class), 20
sosCoordinates, GmlPoint-method (GmlDirectPosition-class), 20
sosCoordinates, GmlPointProperty-method (GmlDirectPosition-class), 20
sosCoordinates, list-method (SOS), 46
sosCoordinates, OmObservation-method (OmObservation-class), 30
sosCoordinates, OmObservationCollection-methods (SOS), 46
sosCoordinates, OmObservationCollection, 33
sosCoordinates, SaSamplingPoint-method (SA), 43
sosCoordinates, SensorML-method (SML), 45
sosCoordinates, SosObservationOffering-method (SOS), 46
sosCoordinates-method (SOS), 46
sosCreate, 55
sosCreateBBOX (sosCreate), 55
sosCreateBBOX, numeric, numeric, numeric, numeric-method (sosCreate), 55
sosCreateBBoxMatrix (sosCreate), 55
sosCreateBBoxMatrix, numeric, numeric, numeric, numeric-method (sosCreate), 55
sosCreateTimeEventTime (sosCreate), 55
sosCreateTimeEventTime, GmlTimeGeometricPrimitive-method (sosCreate), 55
sosCreateTimeEventTime-methods (sosCreate), 55
sosCreateTimeEventTimeList (sosCreate), 55
sosCreateTimeEventTimeList, GmlTimeGeometricPrimitive-method (sosCreate), 55
sosCreateTimeEventTimeList-methods (sosCreate), 55
sosCreateFeatureOfInterest (sosCreate), 55
sosCreateFeatureOfInterest, ANY-method (sosCreate), 55
sosCreateFeatureOfInterest-methods (sosCreate), 55
sosCreateTime (sosCreate), 55
sosCreateTime, SOS, character-method (sosCreate), 55
sosCreateTime-methods (sosCreate), 55
sosCreateTimeInstant (sosCreate), 55
sosCreateTimeInstant, SOS, POSIXt-method (sosCreate), 55
sosCreateTimeInstant-methods (sosCreate), 55
sosDataFieldConverters (SOS), 46
sosDataFieldConverters, SOS-method (SOS), 46
sosDataFieldConvertingFunctions (Defaults), 6
sosDefault (Defaults), 6
sosDefaultCharacterEncoding (Defaults),
<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>(OmObservation-class), 30</td>
<td></td>
</tr>
<tr>
<td>sosResult, OmObservationCollection-method (OmObservationCollection), 33</td>
<td></td>
</tr>
<tr>
<td>sosResult, OmObservationProperty-method (OmObservation-class), 30</td>
<td></td>
</tr>
<tr>
<td>sosResult, OwsExceptionReport-method (OWS), 34</td>
<td></td>
</tr>
<tr>
<td>sosResultModelName (Constants), 6</td>
<td></td>
</tr>
<tr>
<td>sosResultModels (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosResultModels, OwsOperation-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosResultModels, SOS-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosResultModels, SosObservationOffering-method (SosObservationOffering-class), 62</td>
<td></td>
</tr>
<tr>
<td>sosResultModels-methods (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosService (Constants), 6</td>
<td></td>
</tr>
<tr>
<td>sosServiceIdentification (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosServiceIdentification, SOS-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosServiceIdentification-methods (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosServiceProvider (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosServiceProvider, SOS-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosServiceProvider-methods (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosSrsName (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosSrsName, GmlDirectPosition-method (GmlDirectPosition-class), 20</td>
<td></td>
</tr>
<tr>
<td>sosSrsName, GmlPoint-method (GmlDirectPosition-class), 20</td>
<td></td>
</tr>
<tr>
<td>sosSrsName, SOS-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosSrsName-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosSrsName-methods (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>SosSupported (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedComparisonOperators (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedConnectionMethods (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedConnectionMethods (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedGeometryOperands (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedGeometryOperands (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedOperations (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedResponseFormats (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedResponseModes (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedResultModels (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedServiceVersions (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedSpatialOperators (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>SosSupportedTemporalOperators (Supported), 65</td>
<td></td>
</tr>
<tr>
<td>sosSwitchCoordinates (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosSwitchCoordinates, SOS-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosSwitchCoordinates-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosTime (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosTime, GmlTimeInstant-method (GmlDirectPosition-class), 20</td>
<td></td>
</tr>
<tr>
<td>sosTime, GmlTimeInstantProperty-method (GmlDirectPosition-class), 20</td>
<td></td>
</tr>
<tr>
<td>sosTime, GmlTimePeriod-method (GmlDirectPosition-class), 20</td>
<td></td>
</tr>
<tr>
<td>sosTime, GmlTimePosition-method (GmlDirectPosition-class), 20</td>
<td></td>
</tr>
<tr>
<td>sosTime, list-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosTime, SOS-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosTime, SosObservationOffering-method (SosObservationOffering-class), 62</td>
<td></td>
</tr>
<tr>
<td>sosTime-methods (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosTimeFormat (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosTimeFormat, SOS-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosTimeFormat-methods (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosTimeName (Constants), 6</td>
<td></td>
</tr>
<tr>
<td>sosTitle (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosTitle, OwsServiceIdentification-method (OWS), 34</td>
<td></td>
</tr>
<tr>
<td>sosTitle, SOS-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosTitle-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosUOM (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosUOM, data.frame-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosUOM, GmlMeasure-method (GmlDirectPosition-class), 20</td>
<td></td>
</tr>
<tr>
<td>sosUOM, list-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosUOM, OmMeasurement-method (OmMeasurement), 28</td>
<td></td>
</tr>
<tr>
<td>sosUOM, OmObservation-method (OmObservation-class), 30</td>
<td></td>
</tr>
<tr>
<td>sosUOM, OmObservationCollection-method (OmObservationCollection), 33</td>
<td></td>
</tr>
<tr>
<td>sosUrl (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosUrl, SOS-method (SOS), 46</td>
<td></td>
</tr>
<tr>
<td>sosUrl, SOS_1.0.0-method (SOS), 46</td>
<td></td>
</tr>
</tbody>
</table>
sosUrl-methods (SOS), 46
sosVersion (SOS), 46
sosVersion, SOS-method (SOS), 46
sosVersion-methods (SOS), 46
summary.GmlTimePeriod
  (GmlDirectPosition-class), 20
summary.OmObservation
  (OmObservation-class), 30
summary.OmObservationCollection
  (OmObservationCollection), 33
summary.OwsRange (OWS), 34
summary.SOS (SOS), 46
summary.SosObservationOffering (SOS), 46
Supported, 65
SWE, 66
sweBooleanName (Constants), 6
sweCategoryName (Constants), 6
sweCodeSpaceName (Constants), 6
sweComponentName (Constants), 6
SweCompositePhenomenon (SWE), 66
SweCompositePhenomenon-class (SWE), 66
sweCompositePhenomenonName (Constants), 6
sweCoordinateName (Constants), 6
sweCountName (Constants), 6
sweDataArrayName (Constants), 6
sweDataRecordName (Constants), 6
sweEncodingName (Constants), 6
sweFieldName (Constants), 6
sweLocationName (Constants), 6
SwePhenomenon, 68
SwePhenomenon (SWE), 66
SwePhenomenon-class (SWE), 66
SwePhenomenonOrNULL, 68
SwePhenomenonOrNULL-class (SWE), 66
SwePhenomenonProperty (SWE), 66
SwePhenomenonProperty-class (SWE), 66
SwePhenomenonPropertyOrNULL, 68
SwePhenomenonPropertyOrNULL-class (SWE), 66
swePositionName (Constants), 6
sweQuantityName (Constants), 6
sweSimpleDataRecordName (Constants), 6
SweTextBlock (SWE), 66
SweTextBlock-class (SWE), 66
sweTextBlockName (Constants), 6
sweTextName (Constants), 6
sweTimeName (Constants), 6
sweUomName (Constants), 6
sweValueName (Constants), 6
sweValuesName (Constants), 6
sweVectorName (Constants), 6
TM_After (TM_Operators), 68
TM_After-class (TM_Operators), 68
TM_Before (TM_Operators), 68
TM_Before-class (TM_Operators), 68
TM_During (TM_Operators), 68
TM_During-class (TM_Operators), 68
TM_Equals (TM_Operators), 68
TM_Equals-class (TM_Operators), 68
TM_Operators, 68
TM_Operators-class (TM_Operators), 68
toString, GmlDirectPosition-method
  (GmlDirectPosition-class), 20
toString, GmlEnvelope-method
  (GmlDirectPosition-class), 20
toString, GmlFeatureCollection-method
  (GmlDirectPosition-class), 20
toString, GmlFeatureProperty-method
  (GmlDirectPosition-class), 20
toString, GmlGeometry-method
  (GmlDirectPosition-class), 20
toString, GmlMeasure-method
  (GmlDirectPosition-class), 20
toString, GmlPoint-method
  (GmlDirectPosition-class), 20
toString, GmlPointProperty-method
  (GmlDirectPosition-class), 20
toString, GmlTimeInstant-method
  (GmlDirectPosition-class), 20
toString, GmlTimeInstantProperty-method
  (GmlDirectPosition-class), 20
toString, GmlTimeInterval-method
  (GmlDirectPosition-class), 20
toString, GmlTimePeriod-method
  (GmlDirectPosition-class), 20
toString, GmlTimePosition-method
  (GmlDirectPosition-class), 20
toString, OgcBBOX-method (OGC), 26
toString, OgcContains-method (OGC), 26
toString, OgcIntersects-method (OGC), 26
toString, OgcOverlaps-method (OGC), 26
toString, OmMeasurement-method
  (OmMeasurement), 28
toString_OmObservation-method (OmObservation-class), 30
toString_OmObservationCollection-method (OmObservationCollection), 33
toString_OmObservationProperty-method (OmObservation-class), 30
toString_OwsCapabilities-method (OWS), 34
toString_OwsCapabilities_1.1.0-method (OWS), 34
toString_OwsCapabilities_2.0.0-method (OWS), 34
toString_OwsContents-method (OWS), 34
toString_OwsException-method (OWS), 34
toString_OwsExceptionReport-method (OWS), 34
toString_OwsGetCapabilities-method (OWS), 34
toString_OwsGetCapabilities_1.1.0-method (OWS), 34
toString_OwsGetCapabilities_2.0.0-method (OWS), 34
toString_OwsOperation-method (OWS), 34
toString_OwsOperationsMetadata-method (OWS), 34
toString_OwsRange-method (OWS), 34
toString_OwsServiceIdentification-method (OWS), 34
toString_OwsServiceOperation-method (OWS), 34
toString_OwsServiceProvider-method (OWS), 34
toString_SaSamplingPoint-method (SA), 43
toString_SaSamplingSurface-method (SA), 43
toString_SensorML-method (SML), 45
toString_SOS-method (SOS), 46
toString_SOS_1.0.0-method (SOS), 46
toString_SosContents-method (SosContents-class), 53
toString_SosDescribeSensor-method (DescribeSensor), 9
toString_SosEventTime-method (SosEventTime), 58
toString_SosFeatureOfInterest-method (SosFeatureOfInterest-class), 60
toString_SosFilter_Capabilities-method (SosFilter_Capabilities-class), 61
toString_SosGetObservation-method (GetObservation), 16
toString_SosGetObservationById-method (GetObservation), 16
toString_SosObservationOffering-method (SosObservationOffering-class), 62
toString_SweCompositePhenomenon-method (SWE), 66
toString_SwePhenomenon-method (SWE), 66
toString_SwePhenomenonProperty-method (SWE), 66
toString_SweTextBlock-method (SWE), 66
toString_TM_After-method (TM_Operators), 68
toString_TM_Before-method (TM_Operators), 68
toString_TM_During-method (TM_Operators), 68
toString_TM_Equals-method (TM_Operators), 68
xmlInternalDocumentName (Constants), 6
xmlTextNodeName (Constants), 6