Package ‘ows4R’

September 1, 2023

Version  0.3-6
Date    2023-09-01
Title   Interface to OGC Web-Services (OWS)
Maintainer Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Depends R (>= 3.3.0)
Imports methods, R6, httr, openssl, keyring, XML (>= 3.96-1.1),
     geometa (>= 0.7-1), sf, terra, parallel
Suggests testthat, knitr, markdown
Description Provides an Interface to Web-Services defined as standards by the Open Geospatial Con-
     sortium (OGC), including Web Feature Service
     (WFS) for vector data, Web Coverage Service (WCS), Catalogue Ser-
     vice (CSW) for ISO/OGC metadata, Web Processing Service (WPS) for data processes,
     and associated standards such as the common web-service specification (OWS) and OGC Fil-
     ter Encoding. Partial support is provided for the Web Map
     Service (WMS). The purpose is to add support for additional OGC service stan-
     dards such as Web Coverage Processing Service (WCPS), the Sensor
     Observation Service (SOS), or even new standard services emerging such OGC API or Sensor-
     Things.
License  MIT + file LICENSE
URL     https://github.com/eblondel/ows4R,
     https://eblondel.github.io/ows4R/,
     https://www.ogc.org/standards/
BugReports https://github.com/eblondel/ows4R/issues
LazyLoad yes
RoxygenNote  7.2.3
VignetteBuilder knitr
NeedsCompilation no
Author   Emmanuel Blondel [aut, cre] (<https://orcid.org/0000-0002-5870-5762>),
     Alexandre Bennici [ctb] (<https://orcid.org/0000-0003-2160-3487>),
     Norbert Billet [ctb]
Repository  CRAN
Date/Publication  2023-09-01 10:10:02 UTC

R topics documented:

And ................................................................. 4
BBOX ............................................................... 5
BinaryComparisonOpType ........................................... 6
BinaryLogicOpType .................................................. 7
CASClient ......................................................... 9
CSWCapabilities .................................................... 10
CSWClient ......................................................... 11
CSWConstraint ..................................................... 17
CSWDescribeRecord ............................................... 18
CSWGetRecordById ............................................... 20
CSWGetRecords .................................................... 21
CSWHarvest ....................................................... 23
CSWQuery ......................................................... 25
CSWRecordProperty ............................................... 27
CSWTransaction ................................................... 29
getOWSNamespace .................................................. 30
getOWSNamespaces ............................................... 31
Not ................................................................. 31
OGCAbstractObject ............................................... 32
OGCExpression .................................................... 36
OGCFilter .......................................................... 37
Or ................................................................. 39
ows4R ............................................................. 40
OWSBoundingBox ................................................... 40
OWSCapabilities ................................................... 42
OWSClient ......................................................... 45
OWSCodeType ..................................................... 48
OWSGetCapabilities ............................................... 49
OWSHttpRequest ................................................... 50
OWSNamespace ..................................................... 53
OWSOperation ...................................................... 55
OWSOperationsMetadata ........................................... 56
OWSRequest ....................................................... 57
OWSServiceIdentification .......................................... 59
OWSServiceProvider ............................................... 61
OWSUtils .......................................................... 62
OWSWG84BoundingBox .............................................. 63
PropertyIsBetween .................................................. 64
PropertyIsEqualTo .................................................. 66
PropertyIsGreaterThan ............................................. 67
PropertyIsGreaterThanOrEqualTo ................................ 68
PropertyIsLessThan ............................................... 69
R topics documented:

PropertyIsLessThanOrEqualTo ........................................ 70
PropertyIsLike .......................................................... 71
PropertyIsNotEqualTo ..................................................... 73
PropertyIsNull .............................................................. 74
registerOWSNamespace ...................................................... 75
setOWSNamespaces .......................................................... 75
UnaryLogicOpType .......................................................... 76
WCSCapabilities ............................................................. 77
WCSClient ................................................................. 79
WCSCoverage ............................................................... 82
WCSCoverageDescription ................................................. 84
WCSCoverageDomain ........................................................ 86
WCSCoverageFilenameHandler ............................................ 87
WCSCoverageSpatialDomain ................................................. 88
WCSCoverageSummary ....................................................... 90
WCSCoverageTemporalDomain ............................................. 93
WCSDescribeCoverage ....................................................... 95
WCSGetCoverage ........................................................... 97
WCSGridCRS ............................................................... 99
WCSGSElevationDomain .................................................... 100
WCSGSTimeDomain ........................................................ 101
WFSCapabilities ........................................................... 102
WFSClient ................................................................. 104
WFSDescribeFeatureType .................................................. 106
WFSFeatureType ............................................................ 108
WFSFeatureTypeElement ..................................................... 111
WFSGetFeature ............................................................ 113
WMSCapabilities ........................................................... 115
WMSClient ................................................................. 117
WMSGetFeatureInfo .......................................................... 120
WMSLayer ................................................................. 122
WPSBoundingBoxData ....................................................... 126
WPSCapabilities ........................................................... 127
WPSClient ................................................................. 129
WPSComplexData ........................................................... 132
WPSComplexInputDescription ............................................ 134
WPSComplexOutputDescription ........................................... 135
WPSDescribeProcess ......................................................... 136
WPSDescriptionParameter .................................................. 138
WPSException ............................................................. 139
WPSExecute ............................................................... 140
WPSEexecuteResponse ....................................................... 142
WPSFormat ............................................................... 145
WPSInput ................................................................. 147
WPSInputDescription ....................................................... 148
WPSLiteralData ............................................................. 149
WPSLiteralInputDescription ............................................... 151
WPSLiteralOutputDescription ............................................. 152
Description

And

And

Format

R6Class object.

Value

Object of R6Class for modelling an And operator

Super classes

ows4R::OGCAbstractObject -> ows4R::OGExpression -> ows4R::BinaryLogicOpType -> And

Methods

Public methods:

• And$new()
• And$clone()

Method new(): Initializes an And expression

Usage:
And$new(...)

Arguments:
... list of objects of class OGCExpression

Method clone(): The objects of this class are cloneable with this method.

Usage:
And$clone(deep = FALSE)

Arguments:
deep  Whether to make a deep clone.
Examples

```r
expr1 <- PropertyIsEqualTo$new(PropertyName = "property1", Literal = "value1")
expr2 <- PropertyIsEqualTo$new(PropertyName = "property2", Literal = "value2")
and <- And$new(expr1, expr2)
and_xml <- and$encode() # see how it looks like in XML
```

BBOX

Description

BBOX

Format

R6Class object.

Value

Object of R6Class for modelling a BBOX

Super classes

ows4R::OGCAbstractObject -> ows4R::OGCExpression -> BBOX

Public fields

PropertyName property name field for XML encoding
Envelope envelope as object of class GMLEnvelope from geometa

Methods

Public methods:

- BBOX$new()
- BBOX$clone()

Method new(): Initializes a BBOX expression

Usage:

BBOX$new(bbox, srsName = NULL)

Arguments:
**bbox**  an object of class **matrix**

**srsName**  srs name

**Method**  `clone()`: The objects of this class are cloneable with this method.

**Usage:**

```r
BBOX$clone(deep = FALSE)
```

**Arguments:**

deep  Whether to make a deep clone.

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

**Examples**

```r
bbox <- OWSUtils$toBBOX(-180,-90,180,90)
expr <- BBOX$new(bbox)
expr_xml <- expr$encode()  # see how it looks like in XML
```

---

**Description**

BinaryComparisonOpType

**Value**

Object of **R6Class** for modelling an BinaryComparisonOpType

**Super classes**

ows4R::OGCAbstractObject -> ows4R::OGCExpression -> BinaryComparisonOpType

**Public fields**

PropertyName  property name field for XML encoding

Literal  literal field for XML encoding

attrs  attributes for XML encoding
Methods

Public methods:

- `BinaryComparisonOpType$new()`
- `BinaryComparisonOpType$clone()`

**Method new():** Initializes an object extending `BinaryComparisonOpType`

**Usage:**

`BinaryComparisonOpType$new(
  element,
  namespacePrefix,
  PropertyName,
  Literal,
  matchCase = NA
)`

**Arguments:**

- `element` element name
- `namespacePrefix` XML namespace prefix
- `PropertyName` property name
- `Literal` literal
- `matchCase` match case

**Method clone():** The objects of this class are cloneable with this method.

**Usage:**

`BinaryComparisonOpType$clone(deep = FALSE)`

**Arguments:**

- `deep` Whether to make a deep clone.

**Note**

abstract super class of all the property operation classes

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
BinaryLogicOpType

Format

R6Class object.

Value

Object of R6Class for modelling an BinaryLogicOpType

Super classes

ows4R::OGCAbstractObject -> ows4R::OGCExpression -> BinaryLogicOpType

Public fields

operations a list OGC expressions

Methods

Public methods:

• BinaryLogicOpType$new()
• BinaryLogicOpType$setExprVersion()
• BinaryLogicOpType$clone()

Method new(): Initializes a BinaryLogicOpType expression

Usage:
BinaryLogicOpType$new(..., element, namespacePrefix, exprVersion = "1.1.0")

Arguments:
... list of objects of class OGCExpression
element element
namespacePrefix namespacePrefix
exprVersion OGC expression version. Default is "1.1.0"

Method setExprVersion(): Sets expression version. The methods will control that expression versions are set for each of the operations specified in the expression.

Usage:
BinaryLogicOpType$setExprVersion(exprVersion)

Arguments:
exprVersion OGC expression version

Method clone(): The objects of this class are cloneable with this method.

Usage:
BinaryLogicOpType$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Note

abstract super class of all the binary logical operation classes
**Description**

CASClient

**Format**

R6Class object.

**Value**

Object of R6Class with methods for interfacing a Central Authentication Service (CAS).

**Methods**

**Public methods:**

- CASClient$new()
- CASClient$getUrl()
- CASClient$login()
- CASClient$logout()
- CASClient$clone()

**Method new():** Initializes an object of class CASClient

*Usage:*

CASClient$new(url)

*Arguments:*

url base URL of the Central Authentication Service (CAS)

**Method getUrl():** Get CAS base URL

*Usage:*

CASClient$getUrl()

*Returns:*

the base URL

**Method login():** Logs in the CAS

*Usage:*

CASClient$login(user, pwd)

*Arguments:*

user user

pwd password

*Returns:*

TRUE if logged in, FALSE otherwise
**Method logout():** Logs out from the CAS

*Usage:*  
`CASClient$logout()`  

*Returns:* TRUE if logged out, FALSE otherwise

**Method clone():** The objects of this class are cloneable with this method.

*Usage:*  
`CASClient$clone(deep = FALSE)`  

*Arguments:*  
deep Whether to make a deep clone.

**Note**  
Class used internally by *ows4R*

**Author(s)**  
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**Description**  
CSWCapabilities

**Format**  
*R6Class* object.

**Value**  
Object of *R6Class* with methods for interfacing an OGC Catalogue Service for the Web (CSW) Get Capabilities document.

**Super classes**  
`ows4R::OGCAbstractObject` -> `ows4R::OWSCapabilities` -> CSWCapabilities
Methods

**Public methods:**
- `CSWCapabilities$new()`
- `CSWCapabilities$clone()`

**Method new():** Initializes a `CSWCapabilities` object

*Usage:*

```r
CSWCapabilities$new(url, version, client = NULL, logger = NULL, ...)
```

*Arguments:*
- `url`
- `version`
- `client` object of class `CSWClient`
- `logger` logger type: `NULL`, "INFO" or "DEBUG"
- `...` any other parameter to pass to `OWSGetCapabilities` service request

**Method clone():** The objects of this class are cloneable with this method.

*Usage:*

```r
CSWCapabilities$clone(deep = FALSE)
```

*Arguments:*
- `deep` Whether to make a deep clone.

**Note**

Class used to read a `CSWCapabilities` document. The use of `CSWClient` is recommended instead to benefit from the full set of capabilities associated to a CSW server.

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**Description**

`CSWClient`

**Format**

`R6Class` object.

**Value**

Object of `R6Class` with methods for interfacing an OGC Catalogue Service for the Web.
Super classes

ows4R::OGCAbstractObject -> ows4R::OWSClient -> CSWClient

Methods

Public methods:

- CSWClient$new()
- CSWClient$getCapabilities()
- CSWClient$reloadCapabilities()
- CSWClient$describeRecord()
- CSWClient$getRecordById()
- CSWClient$getRecords()
- CSWClient$transaction()
- CSWClient$insertRecord()
- CSWClient$updateRecord()
- CSWClient$deleteRecord()
- CSWClient$deleteRecordById()
- CSWClient$harvestRecord()
- CSWClient$harvestNode()
- CSWClient$clone()

Method new(): This method is used to instantiate a CSWClient with the url of the OGC service. Authentication is supported using basic auth (using user/pwd arguments), bearer token (using token argument), or custom (using headers argument). By default, the logger argument will be set to NULL (no logger). This argument accepts two possible values: INFO: to print only ows4R logs, DEBUG: to print more verbose logs

Usage:

CSWClient$new(
  url,
  serviceVersion = NULL,
  user = NULL,
  pwd = NULL,
  token = NULL,
  headers = c(),
  config = httr::config(),
  cas_url = NULL,
  logger = NULL
)

Arguments:

url url

serviceVersion CSW service version

user user

pwd password

token token
Method `getCapabilities()`: Get CSW capabilities

*Usage:*

\`
CSWClient$getCapabilities()
\`

*Returns:* an object of class `CSWCapabilities`

Method `reloadCapabilities()`: Reloads CSW capabilities

*Usage:*

\`
CSWClient$reloadCapabilities()
\`

Method `describeRecord()`: Describe records. Retrieves the XML schema for CSW records. By default, returns the XML schema for the CSW records (http://www.opengis.net/cat/csw/2.0.2). For other schemas, specify the `outputSchema` required, e.g. http://www.isotc211.org/2005/gmd for ISO 19115/19139 schema

*Usage:*

\`
CSWClient$describeRecord(namespace, ...)
\`

*Arguments:*

namespace `namespace`

... any other parameter to pass to the `CSWDescribeRecord` service request

*Returns:* the service record description

Method `getRecordById()`: Get a record by Id. By default, the record will be returned following the CSW schema (http://www.opengis.net/cat/csw/2.0.2). For other schemas, specify the `outputSchema` required, e.g. http://www.isotc211.org/2005/gmd for ISO 19115/19139 records. The parameter `elementSetName` should among values "full", "brief", "summary". The default "full" corresponds to the full metadata sheet returned. "brief" and "summary" will contain only a subset of the metadata content.

*Usage:*

\`
CSWClient$getRecordById(id, elementSetName = "full", ...)
\`

*Arguments:*

id `record id`

elementSetName `element set name`. Default is "full"

... any other parameter to pass to `CSWGetRecordById` service request

*Returns:* the fetched record, NULL otherwise

Method `getRecords()`: Get records based on a query, object of class `CSWQuery`. The maximum number of records can be set either for the full query (`maxRecords`) or per request (`maxRecordsPerRequest`, default set to 10 records) considering this operation is paginated. By default, the record will be returned following the CSW schema (http://www.opengis.net/cat/csw/2.0.2). For other schemas, specify the `outputSchema` required, e.g. http://www.isotc211.org/2005/gmd for ISO 19115/19139 records.
Usage:
CSWClient$getRecords(
    query = CSWQuery$new(),
    maxRecords = NULL,
    maxRecordsPerRequest = 10L,
    ...
)

Arguments:
query an object of class CSWQuery. By default, an empty query is set.
maxRecords max number of total records. Default is NULL meaning all records are returned.
maxRecordsPerRequest max number of records to return per request. Default set to 10.
... any other parameter to be passed to CSWGetRecords service request

Returns: the list of records. By default each record will be returned as Dublin Core list object. In case ISO 19115/19139 is set as outputSchema, each record will be object of class ISOMetadata from geometa.

Method transaction(): Generic transaction method. Used for inserting, updating or deleting metadata using the transactional CSW service. The type gives the type of transaction (Insert, Update, or Delete). The record

Usage:
CSWClient$transaction(
    type,
    record = NULL,
    recordProperty = NULL,
    constraint = NULL,
    ...
)

Arguments:
type of transaction either "Insert", "Update" or "Delete"
record the record subject of the transaction
recordProperty record property, object of class CSWRecordProperty
constraint constraint, object of class CSWConstraint
... any other parameter to pass to CSWTransaction service request

Returns: TRUE if transaction succeeded, FALSE otherwise

Method insertRecord(): Inserts a new record

Usage:
CSWClient$insertRecord(record, ...)

Arguments:
record record subject of the Insertion
... any other parameter to pass to the transaction

Returns: TRUE if insertion succeeded, FALSE otherwise

Method updateRecord(): Updates an existing record
Usage:
CSWClient$updateRecord(
    record = NULL,
    recordProperty = NULL,
    constraint = NULL,
    ...
)

Arguments:
record record subject of the Insertion
recordProperty record property, object of class CSWRecordProperty
constraint constraint, object of class CSWConstraint
... any other parameter to pass to the transaction

Returns: TRUE if update succeeded, FALSE otherwise

Method deleteRecord(): Deletes an existing (set of) record(s). A constraint (object of class CSWConstraint) can be specified to limit the deletion to some records.

Usage:
CSWClient$deleteRecord(record = NULL, constraint = NULL, ...)

Arguments:
record record subject of the Insertion
constraint constraint, object of class CSWConstraint
... any other parameter to pass to the transaction

Returns: TRUE if deletion succeeded, FALSE otherwise

Method deleteRecordById(): Deletes an existing record by identifier (constraint used to identify the record based on its identifier).

Usage:
CSWClient$deleteRecordById(id)

Arguments:
id record id

Returns: TRUE if deletion succeeded, FALSE otherwise

Method harvestRecord(): Harvests a single record from a sourceUrl, given a resourceType (by default "http://www.isotc211.org/2005/gmd").

Usage:
CSWClient$harvestRecord(
    sourceUrl,
    resourceType = "http://www.isotc211.org/2005/gmd"
)

Arguments:
sourceUrl source URL
resourceType resource type. Default is "http://www.isotc211.org/2005/gmd"

Returns: TRUE if harvesting succeeded, FALSE otherwise
Method `harvestNode()`: Harvests a CSW node (having its endpoint defined by an `url`). A query (object of class `CSWQuery`) can be specified if needed to restrain the harvesting to a subset. The `resourceType` defines the type of resources to be harvested (by default "http://www.isotc211.org/2005/gmd")

Usage:
```r
CSWClient$harvestNode(
  url, 
  query = CSWQuery$new(), 
  resourceType = "http://www.isotc211.org/2005/gmd", 
  sourceBaseUrl 
)
```

Arguments:
- `url` CSW node URL
- `query` a CSW query, object of class `CSWQuery`
- `resourceType` resource type. Default is "http://www.isotc211.org/2005/gmd"
- `sourceBaseUrl` source base URL

Returns: an object of class `list` giving the number of records found and those actually harvested

Method `clone()`: The objects of this class are cloneable with this method.

Usage:
```r
CSWClient$clone(deep = FALSE)
```

Arguments:
- `deep` Whether to make a deep clone.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples
```r
## Not run:
# example based on CSW endpoint responding at http://localhost:8000/csw
csw <- CSWClient$new("http://localhost:8000/csw", serviceVersion = "2.0.2")

# get capabilities
caps <- csw$getCapabilities()

# get records
records <- csw$getRecords()

# get record by id
record <- csw$getRecordById("my-metadata-id")

# Advanced examples at https://github.com/eblondel/ows4R/wiki#csw

## End(Not run)
```
### Description

CSWConstraint

CSWConstraint

### Format

**R6Class** object.

### Value

Object of **R6Class** for modelling an CSW Constraint

### Super class

ows4R::OGCAbstractObject -> CSWConstraint

### Public fields

- **wrap**  internal property for object XML encoding
- **CqlText**  text to use as CQL filter
- **filter**

### Methods

**Public methods:**

- CSWConstraint$new()
- CSWConstraint$setServiceVersion()
- CSWConstraint$clone()

**Method new():**  Initializes a CSWConstraint object to be used to constrain CSW operations.

**Usage:**

```r
CSWConstraint$new(cqlText = NULL, filter = NULL, serviceVersion = "2.0.2")
```

**Arguments:**

- **cqlText**  cqlText, object of class character
- **filter**  filter, object extending **OGCFilter**
- **serviceVersion**  CSW service version. Default is "2.0.2"

**Method setServiceVersion():**  Set service version. This methods ensures that underlying filter property is properly set with the right OGC filter version.

**Usage:**

```r
CSWConstraint$setServiceVersion(serviceVersion)
```
Arguments:
serviceVersion  service version

Method clone(): The objects of this class are cloneable with this method.

Usage:
CSWConstraint$clone(deep = FALSE)

Arguments:
deep  Whether to make a deep clone.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

filter <- OGCFilter$new( PropertyIsEqual$new("apiso:Identifier", "12345") )
cons <- CSWConstraint$new(filter = filter)
cons_xml <- cons$encode() #how it looks like in XML

CSWDescribeRecord  CSWDescribeRecord

Description
CSWDescribeRecord
CSWDescribeRecord

Format
R6Class  object.

Value
Object of R6Class  for modelling a CSW DescribeRecord request

Super classes
ows4R::OGCAbstractObject <-  ows4R::OWSHttp Request  ->  CSWDescribeRecord
Methods

Public methods:

• CSWDescribeRecord$new()
• CSWDescribeRecord$clone()

Method new(): Initializes a CSWDescribeRecord service request

Usage:
CSWDescribeRecord$new(
  capabilities,
  op,
  url,
  version,
  namespace = NULL,
  user = NULL,
  pwd = NULL,
  token = NULL,
  headers = c(),
  config = httr::config(),
  logger = NULL,
  ...
)

Arguments:
  capabilities an object of class CSWCapabilities
  op object of class OWSOperation as retrieved from capabilities
  url url
  version version
  namespace namespace
  user user
  pwd password
  token token
  headers headers
  config config
  logger logger
  ... any parameter to pass to the service request

Method clone(): The objects of this class are cloneable with this method.

Usage:
CSWDescribeRecord$clone(deep = FALSE)

Arguments:
  deep Whether to make a deep clone.

Note
  Class used internally by ows4R to trigger a CSW DescribeRecord request
Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Description
CSWGetRecordById
CSWGetRecordById

Format
R6Class object.

Value
Object of R6Class for modelling a CSW GetRecordById request

Super classes
ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> CSWGetRecordById

Public fields
Id record Id property for request XML encoding
ElementSetName element set name property for XML encoding

Methods
Public methods:
• CSWGetRecordById$new()
• CSWGetRecordById$clone()

Method new(): Initializes a CSWGetRecordById service request
Usage:
CSWGetRecordById$new(  
capabilities,  
op,  
url,  
serviceVersion = "2.0.2",  
user = NULL,  
pwd = NULL,  
token = NULL,  
headers = headers,  
config = httr::config(),  
)
id,
   elementSetName = "full",
   logger = NULL,
   ...
)

**Arguments:**
capabilities an object of class `CSWCapabilities`
op object of class `OWSOperation` as retrieved from capabilities
url url
serviceVersion serviceVersion. Default is "2.0.2
user user
pwd password
token token
headers headers
config config
id record id
elementSetName element set name. Default is "full"
logger logger
... any parameter to pass to the service request

**Method** `clone()`: The objects of this class are cloneable with this method.

**Usage:**
`CSWGetRecordById$clone(deep = FALSE)`

**Arguments:**
deeplwhether to make a deep clone.

**Note**
Class used internally by `ows4R` to trigger a CSW GetRecordById request

**Author(s)**
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**Description**
CSWGetRecords
CSWGetRecords

**Format**
`R6Class` object.
Value
Object of R6Class for modelling a CSW GetRecords request

Super classes
ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> CSWGetRecords

Public fields
Query query property for request XML encoding

Methods

Public methods:
• CSWGetRecords$new()
• CSWGetRecords$clone()

Method new(): Initializes a CSWGetRecordById service request

Usage:
CSWGetRecords$new(
capabilities,
op,
url,
serviceVersion = "2.0.2",
user = NULL,
pwd = NULL,
token = NULL,
headers = list(),
config = httr::config(),
query = NULL,
logger = NULL,
...
)

Arguments:
capabilities an object of class CSWCapabilities
op object of class OWSOperation as retrieved from capabilities
url url
serviceVersion serviceVersion. Default is "2.0.2
user user
pwd password
token token
headers headers
config config
query object of class CSWQuery
logger logger
any parameter to pass to the service request, such as resultType, startPosition, maxRecords, outputFormat, or outputSchema.

Method `clone()`: The objects of this class are cloneable with this method.

Usage:
```
CSWGetRecords$clone(deep = FALSE)
```

Arguments:
- `deep` Whether to make a deep clone.

Note
Class used internally by **ows4R** to trigger a CSW GetRecords request

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

<table>
<thead>
<tr>
<th>CSWHarvest</th>
<th>CSWHarvest</th>
</tr>
</thead>
</table>

### Description

CSWHarvest

<table>
<thead>
<tr>
<th>CSWHarvest</th>
</tr>
</thead>
</table>

### Format

**R6Class** object.

### Value

Object of **R6Class** for modelling a CSW Harvest request

### Super classes

ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> CSWHarvest

### Public fields

- `Source` source property for request XML encoding
- `ResourceType` resource type property for request XML encoding
- `ResourceFormat` resource format property for request XML encoding
Methods

Public methods:
  • `CSWHarvest$new()`
  • `CSWHarvest$clone()`

Method `new()`: Initializes a `CSWHarvest` service request

Usage:
```r
CSWHarvest$new(
  capabilities, 
  op, 
  url, 
  serviceVersion = "2.0.2", 
  user = NULL, 
  pwd = NULL, 
  token = NULL, 
  headers = list(), 
  config = httr::config(), 
  source = NULL, 
  resourceType = "http://www.isotc211.org/schemas/2005/gmd/", 
  resourceFormat = "application/xml", 
  logger = NULL, 
  ...
)
```

Arguments:
- `capabilities` an object of class `CSWCAPabilities`
- `op` object of class `OWSOperation` as retrieved from capabilities
- `url` url
- `serviceVersion` serviceVersion. Default is "2.0.2"
- `user` user
- `pwd` password
- `token` token
- `headers` headers
- `config` config
- `source` source
- `resourceType` resource type. Default is "http://www.isotc211.org/schemas/2005/gmd/"
- `resourceFormat` resource format. Default is "application/xml"
- `logger` logger
- ... any parameter to pass to the service request

Method `clone()`: The objects of this class are cloneable with this method.

Usage:
```r
CSWHarvest$clone(deep = FALSE)
```

Arguments:
- `deep` Whether to make a deep clone.
Note
Class used internally by ows4R to trigger a CSW Harvest request

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Description
CSWQuery
CSWQuery

Format
R6Class object.

Value
Object of R6Class for modelling an CSW Query

Super class
ows4R::OGCAbstractObject -> CSWQuery

Public fields
ElementSetName  element set name property for request XML encoding
constraint  property for request XML encoding

Methods

Public methods:
• CSWQuery$new()
• CSWQuery$setServiceVersion()
• CSWQuery$clone()

Method new(): This method is used to instantiate an CSWQuery object. The ElementSetName can be either "full" (default), "brief" or "summary". A constraint CSWConstraint can be defined for the query. The typeNames indicates to query (default "csw:Record"). The serviceVersion gives the CSW service version (default "2.0.2")

Usage:
CSWQuery$new(
    elementSetName = "full",
    constraint = NULL,
    typeNames = "csw:Record",
    serviceVersion = "2.0.2"
)

Arguments:
  elementSetName  element set name. Default is "full"
  constraint      object of class CSWConstraint
  typeNames       type names
  serviceVersion  CSW service version

Method setServiceVersion(): Set service version. The methods ensures propery naming of typeNames depending on the service version

Usage:
CSWQuery$setServiceVersion(serviceVersion)

Arguments:
  serviceVersion  service version

Method clone(): The objects of this class are cloneable with this method.

Usage:
CSWQuery$clone(deep = FALSE)

Arguments:
  deep  Whether to make a deep clone.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

#CSWQuery - elementSetName
query_full <- CSWQuery$new()
query_brief <- CSWQuery$new(elementSetName = "brief")
query_summary <- CSWQuery$new(elementSetName = "summary")

#CSWQuery - cqlText with title
cons <- CSWConstraint$new(cqlText = "dc:title like '% ips'")
query <- CSWQuery$new(constraint = cons)

#CSW 2.0.2 - Query - Filter / AnyText
filter <- OGCFilter$new(PropertyIsLike$new("csw:AnyText", "%Physio%"))
cons <- CSWConstraint$new(filter = filter)
query <- CSWQuery$new(constraint = cons)

#CSW 2.0.2 - Query - Filter / AnyText Equal
filter <- OGCFilter$new(PropertyIsEqualTo$new("csw:AnyText", "species")

```r
cons <- CSWConstraint$new(filter = filter)
query <- CSWQuery$new(constraint = cons)

#CSW 2.0.2 - Query - Filter / AnyText And Not
filter <- OGCFilter$new(And$new(
  PropertyIsLike$new("csw:AnyText", "%lorem%"),
  PropertyIsLike$new("csw:AnyText", "%ipsum%"),
  Not$new(
    PropertyIsLike$new("csw:AnyText", "%dolor%")
  )
)
cons <- CSWConstraint$new(filter = filter)
query <- CSWQuery$new(constraint = cons)

#CSW 2.0.2 - Query - Filter / AnyText And nested Or
filter <- OGCFilter$new(And$new(
  PropertyIsEqualTo$new("dc:title", "Aliquam fermentum purus quis arcu"),
  PropertyIsEqualTo$new("dc:format", "application/pdf"),
  Or$new(
    PropertyIsEqualTo$new("dc:type", "http://purl.org/dc/dcmitype/Dataset"),
    PropertyIsEqualTo$new("dc:type", "http://purl.org/dc/dcmitype/service"),
    PropertyIsEqualTo$new("dc:type", "http://purl.org/dc/dcmitype/image"),
    PropertyIsEqualTo$new("dc:type", "http://purl.org/dc/dcmitype/text")
  )
)
cons <- CSWConstraint$new(filter = filter)
query <- CSWQuery$new(elementSetName = "brief", constraint = cons)

#CSW 2.0.2 - Query - Filter / BBOX
bbox <- matrix(c(-180,180,-90,90), nrow = 2, ncol = 2, byrow = TRUE, 
dimnames = list(c("x", "y"), c("min", "max")))
filter <- OGCFilter$new( BBOX$new(bbox = bbox) )
cons <- CSWConstraint$new(filter = filter)
query <- CSWQuery$new(elementSetName = "brief", constraint = cons)
```

---

**CSWRecordProperty**

**Description**

CSWRecordProperty

CSWRecordProperty

**Format**

*R6Class* object.

**Value**

Object of *R6Class* for modelling an CSW RecordProperty
Super class

ows4R::OGCAbstractObject -> CSWRecordProperty

Public fields

wrap  internal property for XML encoding
Name  name property for request XML encoding
Value property for request XML encoding

Methods

Public methods:

• CSWRecordProperty$new()
• CSWRecordProperty$clone()

Method new(): Initializes an object of class CSWRecordProperty

Usage:
CSWRecordProperty$new(name, value, cswVersion = "2.0.2")

Arguments:
name name
value value
cswVersion CSW service version. Default is "2.0.2"

Method clone(): The objects of this class are cloneable with this method.

Usage:
CSWRecordProperty$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

rp <- CSWRecordProperty$new(name = "NAME", value = "VALUE")
rp_xml <- rp$encode() # see how it looks in XML
**CSWTransaction**

---

**CSWTransaction**

---

**Description**

<table>
<thead>
<tr>
<th>CSWTransaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSWTransaction</td>
</tr>
</tbody>
</table>

**Format**

| R6Class object. |

**Value**

Object of **R6Class** for modelling a CSW Transaction request

**Super classes**

| ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> CSWTransaction |

**Methods**

**Public methods:**

- `CSWTransaction$new()`
- `CSWTransaction$clone()`

**Method** `new()`: Initializes a **CSWTransaction** service request

**Usage:**

```r
CSWTransaction$new(
  capabilities,
  op,
  url,
  serviceVersion,
  type,
  user = NULL,
  pwd = NULL,
  token = NULL,
  headers = list(),
  record = NULL,
  recordProperty = NULL,
  constraint = NULL,
  logger = NULL,
  ...
)
```

**Arguments:**

- `capabilities` an object of class **CSWCapabilities**
getOWSNamespace

op object of class OWSOperation as retrieved from capabilities
url url
serviceVersion serviceVersion. Default is "2.0.2
type type of transaction, either "Insert", "Update", or "Delete"
user user
pwd password
token token
headers headers
record record
recordProperty record property
constraint constraint
logger logger
... any parameter to pass to the service request

Method clone(): The objects of this class are cloneable with this method.

Usage:
CSWTransaction$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Note
Class used internally by ows4R to trigger a CSW Transaction request

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Description
getOWSNamespace gets a namespace given its id

Usage
getOWSNamespace(id)

Arguments
id namespace prefix

Author(s)
Emmanuel Blondel, <emmanuel.blondel1@gmail.com>
**getOWSNamespace**

### Examples

```r
getOWSNamespace("GMD")
```

### Description

`getOWSNamespace` gets the list of namespaces registered

### Usage

```r
getOWSNamespace()
```

### Author(s)

Emmanuel Blondel, <emmanuel.blondel1@gmail.com>

### Examples

```r
getOWSNamespace()
```

---

**Not**

### Description

Not

### Format

`R6Class` object.

### Value

Object of `R6Class` for modelling an Not

### Super classes

`ows4R::OGCAbstractObject` -> `ows4R::OGCExpression` -> `ows4R::UnaryLogicOpType` -> `Not`
Methods

Public methods:
  • `Not$new()`
  • `Not$clone()`

Method `new()`:
Initializes an Not expression

Usage:
`Not$new(...)`

Arguments:
... list of objects of class OGCExpression

Method `clone()`:
The objects of this class are cloneable with this method.

Usage:
`Not$clone(deep = FALSE)`

Arguments:
deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```r
expr <- PropertyIsEqualTo$new(PropertyName = "property", Literal = "value")
not <- Not$new(expr)
not_xml <- not$encode() # see how it looks like in XML
```

OGCAbstractObject

Description

OGCAbstractObject

Format

R6Class object.

Value

Object of R6Class for modelling an OGCAbstractObject
Public fields

verbose.info logical property to indicate whether INFO logs have to be displayed
verbose.debug logical property to indicate whether DEBUG logs have to be displayed
loggerType logger type, either NULL, "INFO", or "DEBUG"
wrap internal property for XML encoding
element element used for XML encoding
namespace namespace used for XML encoding
defaults default values to be used for XML encoding
attrs attributes to be used for XML encoding

Methods

Public methods:

• OGCAbstractObject$logger()
• OGCAbstractObject$INFO()
• OGCAbstractObject$WARN()
• OGCAbstractObject$ERROR()
• OGCAbstractObject$new()
• OGCAbstractObject$getClassName()
• OGCAbstractObject$getClass()
• OGCAbstractObject$isFieldInheritedFrom()
• OGCAbstractObject$getNamespaceDefinition()
• OGCAbstractObject$encode()
• OGCAbstractObject$clone()

Method logger(): A basic logger function

Usage:
OGCAbstractObject$logger(type, text)

Arguments:
  type type of logs message.
  text log message text to be displayed

Method INFO(): a basic INFO logger function

Usage:
OGCAbstractObject$INFO(text)

Arguments:
  text log message text to be displayed

Method WARN(): a basic WARN logger function

Usage:
OGCAbstractObject$WARN(text)

Arguments:
text  log message text to be displayed

Method ERROR(): a basic ERROR logger function

Usage:
OGCAbstractObject$ERROR(text)

Arguments:
text  log message text to be displayed

Method new(): Initializes an object extending OGCAbstractObject

Usage:
OGCAbstractObject$new(
  xml = NULL,
  element = NULL,
  namespacePrefix = NULL,
  attrs = list(),
  defaults = list(),
  wrap = FALSE,
  logger = NULL
)

Arguments:
xml  object of class XMLInternalNode-class from XML
element  element name
namespacePrefix  namespace prefix for XML encoding
attrs  list of attributes
defaults  list of default values
wrap  whether XML element has to be wrapped during XML encoding
logger  logger

Method getClassName(): Get class name

Usage:
OGCAbstractObject$getClassName()

Returns: an object of class character

Method getClass(): Get class

Usage:
OGCAbstractObject$getClass()

Returns: an object of class R6Class

Method isFieldInheritedFrom(): Utility to return the parent class in which field is defined

Usage:
OGCAbstractObject$isFieldInheritedFrom(field)

Arguments:
field  field name
Returns: object of class R6Class

Method `getNamespaceDefinition()`:

Usage:

```r
OGCAbstractObject$getNamespaceDefinition(recursive = FALSE)
```

Arguments:

- `recursive` Get all namespace recursively

Returns: the namespace definitions as named list

Method `encode()`:

Encodes as XML. The `addNS` parameter controls the addition of XML namespaces. Extra parameters related to `geometa` objects: `geometa_validate` (TRUE by default) and `geometa_inspire` (FALSE by default) can be used to perform ISO and INSPIRE validation respectively. In that case on object of class `geometa::INSPIREMetadataValidator`, with a proper user API key, should be specified as `geometa_inspireValidator` argument.

Usage:

```r
OGCAbstractObject$encode(
  addNS = TRUE,
  geometa_validate = TRUE,
  geometa_inspire = FALSE,
  geometa_inspireValidator = NULL
)
```

Arguments:

- `addNS` controls the addition of XML namespaces
- `geometa_validate` Relates to `geometa` object ISO validation. Default is TRUE
- `geometa_inspire` Relates to `geometa` object INSPIRE validation. Default is FALSE
- `geometa_inspireValidator` Relates to `geometa` object INSPIRE validation. Default is NULL

Returns: an object of class `XMLInternalNode-class` from `XML`

Method `clone()`:

The objects of this class are cloneable with this method.

Usage:

```r
OGCAbstractObject$clone(deep = FALSE)
```

Arguments:

- `deep` Whether to make a deep clone.

Note

abstract class used by `ows4R`

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
OGCExpression

Description
OGCExpression
OGCExpression

Format

R6Class object.

Value
Object of R6Class for modelling an OGC Expression

Super class

ows4R::OGCAbstractObject -> OGCExpression

Methods

Public methods:
• OGCExpression$new()
• OGCExpression$setExprVersion()
• OGCExpression$getExprVersion()
• OGCExpression$clone()

Method new(): Initializes an object of class OGCExpression

Usage:
OGCExpression$new(
  element,
  namespacePrefix,
  attrs = NULL,
  defaults = NULL,
  exprVersion = "1.1.0"
)

Arguments:
  element element name
  namespacePrefix XML namespace prefix
  attrs attributes
  defaults default values
  exprVersion OGC version for the expression
Method `setExprVersion()`: Sets expression version. The methods will control proper XML namespace prefix setting

*Usage:*

OGCExpression$setExprVersion(exprVersion)

*Arguments:*

exprVersion OGC expression version

Method `getExprVersion()`: Gets expression version

*Usage:*

OGCExpression$getExprVersion()

*Returns:* object of class character

Method `clone()`: The objects of this class are cloneable with this method.

*Usage:*

OGCExpression$clone(deep = FALSE)

*Arguments:*

depth Whether to make a deep clone.

**Note**

abstract class

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**OGCFiler**

**OGCFiler**

**Description**

OGCFiler

**Format**

`R6Class` object.

**Value**

Object of `R6Class` for modelling an OGC Filter

**Super class**

`ows4R::OGCAbstractObject` -> OGCFilter
Public fields

expr  OGC expression

Methods

Public methods:

- OGCFilter$new()
- OGCFilter$setFilterVersion()
- OGCFilter$clone()

Method new(): Initializes an object of class OGCFilter.

Usage:
OGCFilter$new(expr, filterVersion = "1.1.0")

Arguments:
expr  object of class OGCExpression
filterVersion  OGC filter version. Default is "1.1.0"

Method setFilterVersion(): Sets the OGC filter version

Usage:
OGCFilter$setFilterVersion(filterVersion)

Arguments:
filterVersion  OGC filter version

Method clone(): The objects of this class are cloneable with this method.

Usage:
OGCFilter$clone(deep = FALSE)

Arguments:
deep  Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

expr <- PropertyIsEqualTo$new(PropertyName = "property", Literal = "value")
not <- Not$new(expr)
not_xml <- not$encode() # see how it looks like in XML
Description

Or

Or

Format

R6Class object.

Value

Object of R6Class for modelling an Or operator

Super classes

ows4R::OGCAbstractObject -> ows4R::OGCExpression -> ows4R::BinaryLogicOpType -> Or

Methods

Public methods:

• Or$new()
• Or$clone()

Method new(): Initializes an Or expression

Usage:
Or$new(...)

Arguments:
... list of objects of class OGCExpression

Method clone(): The objects of this class are cloneable with this method.

Usage:
Or$clone(deep = FALSE)

Arguments:
depth Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Examples

```r
expr1 <- PropertyIsEqualTo$new(PropertyName = "property1", Literal = "value1")
expr2 <- PropertyIsEqualTo$new(PropertyName = "property2", Literal = "value2")
or <- Or$new(expr1, expr2)
or_xml <- or$encode() # see how it looks like in XML
```

---

ows4R

*Interface to OGC Web-Services*

Description

Provides an Interface to Web-Services defined as standards by the Open Geospatial Consortium (OGC), including Web Feature Service (WFS) for vector data, Web Coverage Service (WCS), Catalogue Service (CSW) for ISO/OGC metadata, and associated standards such as the common web-service specification (OWS) and OGC Filter Encoding. Partial support is provided for the Web Map Service (WMS) and Web Processing Service (WPS). The purpose is to add support for additional OGC service standards such as Web Coverage Processing Service (WCPS) or OGC API.

Author(s)

Emmanuel Blondel <emmanuel.blondell@gmail.com>

---

OWSBoundingBox

*OWSBoundingBox*

Description

OWSBoundingBox

Format

R6Class object.

Value

Object of R6Class for modelling an OGC Bounding Box

Super class

ows4R::OGCAbstractObject -> OWSBoundingBox
Public fields

- **attrs**: attributes to be associated to XML
- **LowerCorner**: lower corner coordinates
- **UpperCorner**: upper corner coordinates

Methods

**Public methods:**

- `OWSBoundingBox$new()`
- `OWSBoundingBox$decode()`
- `OWSBoundingBox$getBBOX()`
- `OWSBoundingBox$clone()`

**Method new():** Initializes an object of class `OWSBoundingBox`

**Usage:**

```r
OWSBoundingBox$new(
  xml = NULL,
  element = NULL,
  namespacePrefix = NULL,
  owsVersion,  # OWS version
  serviceVersion,
  logger = NULL
)
```

**Arguments:**

- `xml`: an object of class `XMLInternalNode-class` to initialize from XML
- `element`: element name
- `namespacePrefix`: namespace prefix
- `owsVersion`: OWS version
- `serviceVersion`: service version
- `logger`: logger

**Method decode():** Decodes an object of class `OWSBoundingBox` from XML

**Usage:**

```r
OWSBoundingBox$decode(xml)
```

**Arguments:**

- `xml`: object of class `XMLInternalNode-class` from XML

**Method getBBOX():** Get BBOX as object of class `bbox` from `sf` package

**Usage:**

```r
OWSBoundingBox$getBBOX()
```

**Returns:** a numeric vector of length four, with xmin, ymin, xmax and ymax values

**Method clone():** The objects of this class are cloneable with this method.

**Usage:**

```r
OWSBoundingBox$clone(deep = FALSE)
```

**Arguments:**

- `deep`: Whether to make a deep clone.
Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

OWSCapabilities

Description
OWSGetCapabilities

Format
R6Class object.

Value
Object of R6Class with methods for interfacing an abstract OWS Get Capabilities document.

Super class
ows4R::OGCAbstractObject -> OWSCapabilities

Methods

Public methods:
• OWSCapabilities$new()
• OWSCapabilities$setClient()
• OWSCapabilities$getClient()
• OWSCapabilities$getUrl()
• OWSCapabilities$getService()
• OWSCapabilities$getServiceVersion()
• OWSCapabilities$getOWSVersion()
• OWSCapabilities$getRequest()
• OWSCapabilities$getServiceIdentification()
• OWSCapabilities$getServiceProvider()
• OWSCapabilities$getOperationsMetadata()
• OWSCapabilities$clone()

Method new(): Initializes a OWSCapabilities object

Usage:
OWSCapabilities$new(
  element = NULL,
  namespacePrefix = NULL,
  url,
  service,
  owsVersion,
  serviceVersion,
  logger = NULL,
  ...
)

Arguments:
  element element
  namespacePrefix namespace prefix
  url url
  service service
  owsVersion OWS version
  serviceVersion service version
  logger logger type NULL, "INFO" or "DEBUG"
  ... any other parameter to pass to OWSGetCapabilities service request

Method setClient(): Sets the OGC client
  Usage:
  OWSCapabilities$setClient(client)
  Arguments:
  client an object extending OWSClient

Method getClient(): Get client
  Usage:
  OWSCapabilities$getClient()
  Arguments:
  an object extending OWSClient

Method getUrl(): Get URL
  Usage:
  OWSCapabilities$getUrl()
  Returns: an object of class character

Method getService(): Get service
  Usage:
  OWSCapabilities$getService()
  Returns: an object of class character

Method getServiceVersion(): Get service version
  Usage:
OWL Capabilities

\texttt{OWSCapabilities\$getServiceVersion()}

\textit{Returns:} an object of class character

\textbf{Method} \texttt{getOWSVersion():} Get OWS version

\textit{Usage:}
\texttt{OWSCapabilities\$getOWSVersion()}

\textit{Returns:} an object of class character

\textbf{Method} \texttt{getRequest():} Get request

\textit{Usage:}
\texttt{OWSCapabilities\$getRequest()}

\textit{Returns:} an object of class \texttt{OWSGetCapabilities}

\textbf{Method} \texttt{getServiceIdentification():} Get service identification

\textit{Usage:}
\texttt{OWSCapabilities\$getServiceIdentification()}

\textit{Returns:} an object of class \texttt{OWSServiceIdentification}

\textbf{Method} \texttt{getServiceProvider():} Get service provider

\textit{Usage:}
\texttt{OWSCapabilities\$getServiceProvider()}

\textit{Returns:} an object of class \texttt{OWSServiceProvider}

\textbf{Method} \texttt{getOperationsMetadata():} Get service operations metadata

\textit{Usage:}
\texttt{OWSCapabilities\$getOperationsMetadata()}

\textit{Returns:} an object of class \texttt{OWSOperationsMetadata}

\textbf{Method} \texttt{clone():} The objects of this class are cloneable with this method.

\textit{Usage:}
\texttt{OWSCapabilities\$clone(deep = FALSE)}

\textit{Arguments:}
- deep Whether to make a deep clone.

\textbf{Note}

abstract class used by \texttt{ows4R}

\textbf{Author(s)}

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
**Description**

OWSClient

**Format**

*R6Class* object.

**Value**

Object of *R6Class* with methods for interfacing a Common OGC web-service.

**Super class**

*ows4R::OGCAbstractObject* -> OWSClient

**Public fields**

- `url`: Base url of the OWS service
- `version`: version of the OWS service
- `capabilities`: object giving the OWS service capabilities

**Methods**

**Public methods:**

- `OWSClient$new()`
- `OWSClient$getUrl()`
- `OWSClient$getVersion()`
- `OWSClient$getCapabilities()`
- `OWSClient$getUser()`
- `OWSClient$getPwd()`
- `OWSClient$getToken()`
- `OWSClient$getHeaders()`
- `OWSClient$getConfig()`
- `OWSClient$getCASUrl()`
- `OWSClient$clone()`

**Method new():** This method is used to instantiate a OWSClient with the `url` of the OGC service. Authentication is supported using basic auth (using `user/pwd` arguments), bearer token (using `token` argument), or custom (using `headers` argument). By default, the logger argument will be set to NULL (no logger). This argument accepts two possible values: `INFO`: to print only *ows4R* logs, `DEBUG`: to print more verbose logs
Usage:
```r
OWSClient$new(
  url,
  service,
  serviceVersion,
  user = NULL,
  pwd = NULL,
  token = NULL,
  headers = c(),
  config = httr::config(),
  cas_url = NULL,
  logger = NULL
)
```

Arguments:
- `url`: url
- `service`: service name
- `serviceVersion`: CSW service version
- `user`: user
- `pwd`: password
- `token`: token
- `headers`: headers
- `config`: config
- `cas_url`: Central Authentication Service (CAS) URL
- `logger`: logger

Method `getUrl()`: Get URL
```r
Usage:
OWSClient$getUrl()
```

Returns: the url of the service, object of class character

Method `getVersion()`: Get version
```r
Usage:
OWSClient$getVersion()
```

Returns: the version of the service, object of class character

Method `getCapabilities()`: Get capabilities
```r
Usage:
OWSClient$getCapabilities()
```

Returns: the capabilities, object of class `OWSCapabilities`

Method `getUser()`: Get user
```r
Usage:
OWSClient$getUser()
```

Returns: the user, object of class character
Method `getPwd()`: Get password

Usage:

```r
OWSClient$getPwd()
```

Returns: the password, object of class `character`

Method `getToken()`: Get token

Usage:

```r
OWSClient$getToken()
```

Returns: the token, object of class `character`

Method `getHeaders()`: Get headers

Usage:

```r
OWSClient$getHeaders()
```

Returns: the headers, object of class `character`

Method `getConfig()`: Get `httr` config

Usage:

```r
OWSClient$getConfig()
```

Returns: the `httr` config, if any

Method `getCASUrl()`: Get CAS URL

Usage:

```r
OWSClient$getCASUrl()
```

Returns: a CAS URL

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```r
OWSClient$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Note

Abstract class used internally by `ows4R`

Author(s)

Emmanuel Blondel <emmanuel.blondell1@gmail.com>
OWSCodeType

Description

OWSCodeType

Format

R6Class object.

Value

Object of R6Class for modelling an OWS CodeType

Super class

ows4R::OGCAbstractObject -> OWSCodeType

Public fields

value code type

Methods

Public methods:

• OWSCodeType$new()
• OWSCodeType$clone()

Method new(): Initializes an object of class OWSCodeType

Usage:

OWSCodeType$new(xml = NULL, owsVersion = "1.1", value)

Arguments:

xml object of class XMLInternalNode-class from XML
owsVersion OWS version. Default is "1.1"
value the code type

Method clone(): The objects of this class are cloneable with this method.

Usage:

OWSCodeType$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Description

OWSGetCapabilities

Format

R6Class object.

Value

Object of R6Class for modelling a GetCapabilities request

Super classes

ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> OWSGetCapabilities

Methods

Public methods:

• OWSGetCapabilities$new()
• OWSGetCapabilities$clone()

Method new(): Initializes an OWSGetCapabilities service request

Usage:

OWSGetCapabilities$new(  
element = NULL,  
namespacePrefix = NULL,  
url,  
service,  
version,  
user = NULL,  
pwd = NULL,  
token = NULL,  
headers = c(),  
config = httr::config(),  
...  
)

Arguments:

element element

namespacePrefix namespace prefix

url url
```

service  service name
version  service version
user     user
pwd      password
token    token
headers  headers
config   config
...      any other parameter to pass to the request

Method clone(): The objects of this class are cloneable with this method.

Usage:
OWSGetCapabilities$clone(deep = FALSE)

Arguments:
deepp Whether to make a deep clone.

Note
Class used internally by ows4R to trigger a OWS GetCapabilities request

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
```

---

**Description**

OWSHttpRequest

**Format**

*R6Class* object.

**Value**

Object of *R6Class* for modelling a generic OWS http request

**Super class**

`ows4R::OGCAbstractObject` -> OWSHttpRequest
Methods

Public methods:

• `OWSHttpRequest$new()`
• `OWSHttpRequest$execute()`
• `OWSHttpRequest$getCapabilities()`
• `OWSHttpRequest$getRequest()`
• `OWSHttpRequest$getRequestHeaders()`
• `OWSHttpRequest$getStatus()`
• `OWSHttpRequest$getResponse()`
• `OWSHttpRequest$getException()`
• `OWSHttpRequest$getResult()`
• `OWSHttpRequest$setResult()`
• `OWSHttpRequest$clone()`

Method `new()`: Initializes an OWS HTTP request

Usage:
```r
OWSHttpRequest$new(
  element,
  namespacePrefix,
  capabilities,
  op,
  type,
  url,
  request,
  user = NULL,
  pwd = NULL,
  token = NULL,
  headers = c(),
  config = httr::config(),
  namedParams = NULL,
  attrs = NULL,
  contentType = "text/xml",
  mimeType = "text/xml",
  logger = NULL,
  ...
)
```

Arguments:
- `element` element
- `namespacePrefix` namespace prefix
- `capabilities` object of class or extending `OWSCapabilities`
- `op` object of class `OWSOperation`
- `type` type of request, eg. GET, POST
- `url` url
- `request` request name
user
pwd
user
password
token
token
headers
headers
config
config
namedParams
a named list
attrs
attributes
contentType
content type. Default value is "text/xml"
mimeType
mime type. Default value is "text/xml"
logger
logger
... any other parameter

**Method** `execute()`: Executes the request

*Usage:*

```java
OWSHttpRequest.execute()
```

**Method** `getCapabilities()`: Get capabilities

*Usage:*

```java
OWSHttpRequest.getCapabilities()
```

*Returns:* an object of class or extending `OWSCapabilities`

**Method** `getRequest()`: Get request

*Usage:*

```java
OWSHttpRequest.getRequest()
```

*Returns:* the request

**Method** `getRequestHeaders()`: Get request headers

*Usage:*

```java
OWSHttpRequest.getRequestHeaders()
```

*Returns:* the request headers

**Method** `getStatus()`: Get status code

*Usage:*

```java
OWSHttpRequest.getStatus()
```

*Returns:* the request status code

**Method** `getResponse()`: Get request response

*Usage:*

```java
OWSHttpRequest.getResponse()
```

*Returns:* the request response

**Method** `getException()`: Get request exception

*Usage:*

```java
OWSHttpRequest.getException()
```
OWSHttpRequest$getException()

**Returns:** the request exception

**Method** getResult(): Get the result TRUE if the request is successful, FALSE otherwise

**Usage:**
OWSHttpRequest$getResult()

**Returns:** the result, object of class logical

**Method** setResult(): Set the result

**Usage:**
OWSHttpRequest$setResult(result)

**Arguments:**
result object of class logical

**Method** clone(): The objects of this class are cloneable with this method.

**Usage:**
OWSHttpRequest$clone(deep = FALSE)

**Arguments:**
deep Whether to make a deep clone.

**Note**
Abstract class used internally by ows4R

**Author(s)**
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**Description**
OWSNamespace

**Format**

R6Class object.

**Value**

Object of R6Class for modelling an OWS Namespace
Public fields

   id  namespace id
   uri namespace uri

Methods

Public methods:

• OWSNamespace$new()
• OWSNamespace$getDefinition()
• OWSNamespace$clone()

Method new(): Initializes an OWSNamespace

Usage:
OWSNamespace$new(id, uri)

Arguments:
   id id
   uri uri

Method getDefinition(): Get namespace definition

Usage:
OWSNamespace$getDefinition()

Returns: a named list with id and uri

Method clone(): The objects of this class are cloneable with this method.

Usage:
OWSNamespace$clone(deep = FALSE)

Arguments:
   deep Whether to make a deep clone.

Note

class used internally by ows4R for specifying XML namespaces

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
**Description**

OWSOperation

**Format**

*R6Class* object.

**Value**

Object of *R6Class* for modelling an OGC Operation

**Methods**

**Public methods:**

- `OWSOperation$new()`
- `OWSOperation$getName()`
- `OWSOperation$getParameters()`
- `OWSOperation$getParameter()`
- `OWSOperation$clone()`

**Method `new()`:** Initializes an object of class `OWSOperation`.

*Usage:*

`OWSOperation$new(xmlObj, owsVersion, serviceVersion)`

*Arguments:*

- `xmlObj` object of class `XMLInternalNode-class` from `XML`
- `owsVersion` OWS version
- `serviceVersion` service version

**Method `getName()`:** Get operation name

*Usage:*

`OWSOperation$getName()`

*Returns:* an object of class character

**Method `getParameters()`:** Get parameters

*Usage:*

`OWSOperation$getParameters()`

*Returns:* the parameters

**Method `getParameter()`:** Get parameter
Usage:
OWSOperation$getParameter(name)

Arguments:
name

Returns: the parameter

Method clone(): The objects of this class are cloneable with this method.

Usage:
OWSOperation$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Note
Internal class used internally by ows4r when reading capabilities documents

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Description
OWSOperationsMetadata

Format
R6Class object.

Value
Object of R6Class for modelling an OGC Operations Metadata

Methods

Public methods:
• OWSOperationsMetadata$new()
• OWSOperationsMetadata$getOperations()
• OWSOperationsMetadata$clone()

Method new(): Initializes an OWSOperationsMetadata object

Usage:
OWSRequest

```
OWSOperationsMetadata$new(xmlObj, owsVersion, serviceVersion)
Arguments:
xmlObj object of class XMLInternalNode-class from XML
owsVersion OWS version
serviceVersion service version

Method getOperations(): Get operations
Usage:
OWSOperationsMetadata$getOperations()
Returns: a list of OWSOperation

Method clone(): The objects of this class are cloneable with this method.
Usage:
OWSOperationsMetadata$clone(deep = FALSE)
Arguments:
deep Whether to make a deep clone.

Note
Abstract class used internally by ows4R

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
```

---

**Description**

OWSRequest
OWSRequest

**Format**

*R6Class* object.

**Value**

Object of *R6Class* modelling a OWS Service Capability Request

**Super class**

*ows4R::OGCAbstractObject* -> OWSRequest
Methods

Public methods:

- `OWSRequest$new()`
- `OWSRequest$getName()`
- `OWSRequest$getFormats()`
- `OWSRequest$clone()`

Method `new()`: Initializes an object of class `OWSRequest`.

Usage:

```
OWSRequest$new(xmlObj, capabilities, version, logger = NULL)
```

Arguments:

- `xmlObj` object of class `XMLInternalNode-class` from `XML`.
- `capabilities` an object of class or extending `OWSCapabilities`.
- `version` version.
- `logger` logger.

Method `getName()`: Get request name.

Usage:

```
OWSRequest$getName()
```

Returns: the name, object of class character.

Method `getFormats()`: Get request formats.

Usage:

```
OWSRequest$getFormats()
```

Returns: the formats, object (vector) of class character.

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
OWSRequest$clone(deep = FALSE)
```

Arguments:

- `deep` Whether to make a deep clone.

Note

Abstract class used by `ows4R`.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Description

OWSServiceIdentification

OWSServiceIdentification

Format

R6Class object.

Value

Object of R6Class for modelling an OGC Service Identification

Methods

Public methods:

- OWSServiceIdentification$new()
- OWSServiceIdentification$getName()
- OWSServiceIdentification$getTitle()
- OWSServiceIdentification$getAbstract()
- OWSServiceIdentification$getKeywords()
- OWSServiceIdentification$getOnlineResource()
- OWSServiceIdentification$getServiceType()
- OWSServiceIdentification$getServiceTypeVersion()
- OWSServiceIdentification$getFees()
- OWSServiceIdentification$getAccessConstraints()
- OWSServiceIdentification$clone()

Method new(): Initializes an object of class OWSServiceIdentification

Usage:
OWSServiceIdentification$new(xmlObj, owsVersion, serviceVersion)

Arguments:
xmlObj object of class XMLInternalNode-class from XML
owsVersion OWS version
serviceVersion service version

Method getName(): Get service identification - name

Usage:
OWSServiceIdentification$getName()
Returns: the name, object of class character

Method getTitle(): Get service identification - title

Usage:
OWSServiceIdentification$getTitle()

Returns: the title, object of class character

Method getAbstract(): Get service identification - abstract

Usage:
OWSServiceIdentification$getAbstract()

Returns: the abstract, object of class character

Method getKeywords(): Get service identification - keywords

Usage:
OWSServiceIdentification$getKeywords()

Returns: the keywords, object of class character

Method getOnlineResource(): Get service identification - online resource

Usage:
OWSServiceIdentification$getOnlineResource()

Returns: the online resource, object of class character

Method getServiceType(): Get service identification - service type

Usage:
OWSServiceIdentification$getServiceType()

Returns: the service type, object of class character

Method getServiceTypeVersion(): Get service identification - service type version

Usage:
OWSServiceIdentification$getServiceTypeVersion()

Returns: the service type version, object of class character

Method getFees(): Get service identification - fees

Usage:
OWSServiceIdentification$getFees()

Returns: the fees, object of class character

Method getAccessConstraints(): Get service identification - access constraints

Usage:
OWSServiceIdentification$getAccessConstraints()

Returns: the access constraints, object of class character

Method clone(): The objects of this class are cloneable with this method.

Usage:
OWSServiceIdentification$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.
**Note**

Abstract class used internally by **ows4R**

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**Description**

**OWSServiceProvider**

**OWSServiceProvider**

**Format**

**R6Class** object.

**Value**

Object of **R6Class** for modelling an OGC Service Provider

**Methods**

**Public methods:**

- `OWSServiceProvider$new()`  
- `OWSServiceProvider$getProviderName()`  
- `OWSServiceProvider$getProviderSite()`  
- `OWSServiceProvider$getServiceContact()`  
- `OWSServiceProvider$clone()`

**Method** `new()`:

Initializes an object of class **OWSServiceProvider**

**Usage:**

```
OWSServiceProvider$new(xmlObj, owsVersion, serviceVersion)
```

**Arguments:**

- `xmlObj` object of class **XMLInternalNode-class** from **XML**
- `owsVersion` OWS version
- `serviceVersion` service version

**Method** `getProviderName()`:

Get provider name

**Usage:**

```
OWSServiceProvider$getProviderName()
```

**Arguments:**
the provider name, object of class character

**Method** `getProviderSite()`: Get provider site

*Usage:*

```r
OWSServiceProvider$getProviderSite()
```

*Arguments:*

- the provider site, object of class character

**Method** `getServiceContact()`: Get provider contact

*Usage:*

```r
OWSServiceProvider$getServiceContact()
```

*Arguments:*

- the provider contact, object of class `ISOResponsibleParty` from `geometa`

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

```r
OWSServiceProvider$clone(deep = FALSE)
```

*Arguments:*

- deep Whether to make a deep clone.

**Note**

Abstract class used internally by `ows4R`

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**Description**

OWSUtils

**Usage**

```r
OWSUtils
```

**Format**

- `R6Class` object.

**Value**

set of OWS Utilities
**Static methods**

getNamespaces() Get the namespaces associated to a given XML object
findNamespace(namespaces, id, uri) Finds a namespace by id or by URI
toBBOX(xmin, xmax, ymin, ymax) Creates a bbox matrix from min/max x/y coordinates
findP4s(srsName, morphToESRI) Finds the PROJ4 string definition for a given srsName
toCRS(srsName) Converts a srsName into a CRS object
toEPSG(crs) Get the EPSG code from a CRS object
getAspectRatio(bbox) Get the aspect ratio for a given bbox

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

**Examples**

```r
#toBBOX
bbox <- OWSUtils$toBBOX(-180,-90,180,90)

#toCRS
crs <- OWSUtils$toCRS("EPSG:4326")
```

**Description**

OWSWGS84BoundingBox
OWSWGS84BoundingBox

**Format**

*R6Class* object.

**Value**

Object of *R6Class* for modelling an OGC WGS84 BoundingBox

**Super classes**

`ows4R::OGCAbstractObject` -> `ows4R::OWSBoundingBox` -> OWSWGS84BoundingBox
Methods

Public methods:

- OWSWGS84BoundingBox$new()
- OWSWGS84BoundingBox$clone()

Method new(): Initializes an object of class OWSBoundingBox

Usage:
OWSWGS84BoundingBox$new(xml = NULL, owsVersion, serviceVersion, logger = NULL)

Arguments:
xml an object of class XMLInternalNode-class to initialize from XML
owsVersion OWS version
serviceVersion service version
logger logger

Method clone(): The objects of this class are cloneable with this method.

Usage:
OWSWGS84BoundingBox$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Note

Class used internally by geometa

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Description

PropertyIsBetween

Format

R6Class object.

Value

Object of R6Class for modelling an PropertyIsBetween
Super classes

ows4R::OGCAbstractObject -> ows4R::OGCExpression -> PropertyIsBetween

Public fields

PropertyName  property name field for XML encoding
lower  lower value
upper  upper value

Methods

Public methods:

• PropertyIsBetween$new()
• PropertyIsBetween$clone()

Method new(): Initializes an object extending PropertyIsLike

Usage:
PropertyIsBetween$new(PropertyName, lower, upper)

Arguments:
PropertyName  property name
lower  lower value
upper  upper value

Method clone(): The objects of this class are cloneable with this method.

Usage:
PropertyIsBetween$clone(deep = FALSE)

Arguments:
deep  Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

expr <- PropertyIsBetween$new(PropertyName = "property", lower = 1, upper = 10)
expr_xml <- expr$encode() #see how it looks like in XML
Description

PropertyIsEqualTo

Format

\texttt{R6Class} object.

Value

Object of \texttt{R6Class} for modelling an PropertyIsEqualTo

Super classes

\texttt{ows4R::OGCAbstractObject -> ows4R::OGCExpression -> ows4R::BinaryComparisonOpType -> PropertyIsEqualTo}

Methods

Public methods:

- \texttt{PropertyIsEqualTo$new()}
- \texttt{PropertyIsEqualTo$clone()}

Method \texttt{new()}: Initializes an object extending PropertyIsEqualTo

Usage:

\texttt{PropertyIsEqualTo$new(PropertyName, Literal, matchCase = NA)}

Arguments:

- \texttt{PropertyName} property name
- \texttt{Literal} literal
- \texttt{matchCase} match case

Method \texttt{clone()}: The objects of this class are cloneable with this method.

Usage:

\texttt{PropertyIsEqualTo$clone(deep = FALSE)}

Arguments:

- \texttt{deep} Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Examples

```r
expr <- PropertyIsEqualTo$new(PropertyName = "property", Literal = "value")
expr_xml <- expr$encode() # see how it looks like in XML
```

Description

PropertyIsGreaterThan

Value

Object of `R6Class` for modelling an PropertyIsGreaterThan

Super classes

`ows4R::OGCAbstractObject` -> `ows4R::OGCExpression` -> `ows4R::BinaryComparisonOpType` -> `PropertyIsGreaterThan`

Methods

Public methods:

- `PropertyIsGreaterThan$new()`
- `PropertyIsGreaterThan$clone()`

Method `new()`: Initializes an object extending `PropertyIsGreaterThan`

Usage:

```r
PropertyIsGreaterThan$new(PropertyName, Literal, matchCase = NA)
```

Arguments:

- `PropertyName` property name
- `Literal` literal
- `matchCase` match case

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```r
PropertyIsGreaterThan$clone(deep = FALSE)
```

Arguments:

- `deep` Whether to make a deep clone.
Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```r
expr <- PropertyIsGreaterThan$new(PropertyName = "property", Literal = "value")
expr_xml <- expr$encode() # see how it looks like in XML
```

Description

PropertyIsGreaterThanOrEqualTo

Format

*R6Class* object.

Value

Object of *R6Class* for modelling an PropertyIsGreaterThanOrEqualTo

Super classes

`ows4R::OGCAbstractObject` - `ows4R::OGCExpression` - `ows4R::BinaryComparisonOpType` -> PropertyIsGreaterThanOrEqualTo

Methods

Public methods:
- PropertyIsGreaterThanOrEqualTo$new()
- PropertyIsGreaterThanOrEqualTo$clone()

Method new(): Initializes an object extending PropertyIsGreaterThanOrEqualTo

Usage:
```
PropertyIsGreaterThanOrEqualTo$new(PropertyName, Literal, matchCase = NA)
```

Arguments:
- `PropertyName` property name
- `Literal` literal
- `matchCase` match case

Method clone(): The objects of this class are cloneable with this method.
Usage:
PropertyIsGreaterThanOrEqualTo$clone(deep = FALSE)

Arguments:
depth Whether to make a deep clone.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples
expr <- PropertyIsGreaterThanOrEqualTo$new(PropertyName = "property", Literal = "value")
expr_xml <- expr$encode() # see how it looks like in XML

Description
PropertyIsLessThan

Format
R6Class object.

Value
Object of R6Class for modelling an PropertyIsLessThan

Super classes
ows4R::OGCAbstractObject -> ows4R::OGCExpression -> ows4R::BinaryComparisonOpType
-> PropertyIsLessThan

Methods

Public methods:
- PropertyIsLessThan$new()
- PropertyIsLessThan$clone()

Method new(): Initializes an object extending PropertyIsLessThan

Usage:
PropertyIsLessThan$new(PropertyName, Literal, matchCase = NA)

Arguments:
PropertyName property name
Literal literal
matchCase match case

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*
`PropertyIsLessThan$clone(deep = FALSE)`

*Arguments:*
depth Whether to make a deep clone.

**Author(s)**
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

**Examples**
```r
expr <- PropertyIsLessThan$new(PropertyName = "property", Literal = "value")
expr_xml <- expr$encode()  # see how it looks like in XML
```

---

**PropertyIsLessThanOrEqualTo**

**Description**
- `PropertyIsLessThanOrEqualTo`
- `PropertyIsLessThanOrEqualTo`

**Format**
- **R6Class** object.

**Value**
Object of **R6Class** for modelling an `PropertyIsLessThanOrEqualTo`

**Super classes**
- `ows4R::OGCAbstractObject` -> `ows4R::OGCExpression` -> `ows4R::BinaryComparisonOpType` -> `PropertyIsLessThanOrEqualTo`
Methods

Public methods:

- `PropertyIsLessThanOrEqualTo$new()`
- `PropertyIsLessThanOrEqualTo$clone()`

Method `new()`:

**Usage:**
`PropertyIsLessThanOrEqualTo$new(PropertyName, Literal, matchCase = NA)`

**Arguments:**
- `PropertyName` property name
- `Literal` literal
- `matchCase` match case

Method `clone()`:

**Usage:**
`PropertyIsLessThanOrEqualTo$clone(deep = FALSE)`

**Arguments:**
- `deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```r
expr <- PropertyIsLessThanOrEqualTo$new(PropertyName = "property", Literal = "value")
expr_xml <- expr$encode() # see how it looks like in XML
```

---

**Description**

PropertyIsLike

PropertyIsLike

**Format**

*R6Class* object.

**Value**

Object of *R6Class* for modelling an PropertyIsLike
Super classes

ows4R::OGCAbstractObject -> ows4R::OGCExpression -> PropertyIsLike

Public fields

PropertyName property name field for XML encoding
Literal literal field for XML encoding
attrs attributes for XML encoding

Methods

Public methods:
• PropertyIsLike$new()
• PropertyIsLike$clone()

Method new(): Initializes an object extending PropertyIsLike

Usage:
PropertyIsLike$new(
  PropertyName,
  Literal,
  escapeChar = "\\",
  singleChar = "_",
  wildCard = "%",
  matchCase = NA
)

Arguments:
PropertyName property name
Literal literal
escapeChar escape character. Default is "\"
singleChar single character. Default is "_"
wildCard wildcard
matchCase match case

Method clone(): The objects of this class are cloneable with this method.

Usage:
PropertyIsLike$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

expr <- PropertyIsLike$new(PropertyName = "property", Literal = "value")
expr_xml <- expr$encode() # see how it looks like in XML
**Description**

PropertyIsNotEqualTo

**Format**

*R6Class* object.

**Value**

Object of *R6Class* for modelling an *PropertyIsNotEqualTo*

**Super classes**

`ows4R::OGCAbstractObject` -> `ows4R::OGCExpression` -> `ows4R::BinaryComparisonOpType` -> `PropertyIsNotEqualTo`

**Methods**

**Public methods:**

- `PropertyIsNotEqualTo$new()`
- `PropertyIsNotEqualTo$clone()`

**Method new():** Initializes an object extending *PropertyIsNotEqualTo*

*Usage:*

`PropertyIsNotEqualTo$new(PropertyName, Literal, matchCase = NA)`

*Arguments:*

- `PropertyName` property name
- `Literal` literal
- `matchCase` match case

**Method clone():** The objects of this class are cloneable with this method.

*Usage:*

`PropertyIsNotEqualTo$clone(deep = FALSE)`

*Arguments:*

- `deep` Whether to make a deep clone.

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Examples

```r
expr <- PropertyIsNotEqualTo$new(PropertyName = "property", Literal = "value")
expr_xml <- expr$encode()  # see how it looks like in XML
```

Description

PropertyIsNull

Format

**R6Class** object.

Value

Object of **R6Class** for modelling an PropertyIsNull

Super classes

```
ows4R::OGCAbstractObject -> ows4R::OGCExpression -> PropertyIsNull
```

Public fields

PropertyName property name field for XML encoding

Methods

**Public methods:**

- `PropertyIsNull$new()`  
- `PropertyIsNull$clone()`

**Method** `new()`:

Initializes an object extending **PropertyIsLike**

**Usage:**

`PropertyIsNull$new(PropertyName)`

**Arguments:**

PropertyName property name

**Method** `clone()`:

The objects of this class are cloneable with this method.

**Usage:**

`PropertyIsNull$clone(deep = FALSE)`

**Arguments:**

deep Whether to make a deep clone.
registerOWSNamespace

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples
expr <- PropertyIsNull$new(PropertyName = "property")
expr_xml <- expr$encode() # see how it looks like in XML

registerOWSNamespace(id, uri, force)

Arguments
id prefix of the namespace
uri URI of the namespace
force logical parameter indicating if registration has to be forced in case the identified namespace is already registered

Author(s)
Emmanuel Blondel, <emmanuel.blondel1@gmail.com>

Examples
registerOWSNamespace(id = "myprefix", uri = "http://someuri")

setOWSNamespaces

Description
setOWSNamespaces

Usage
setOWSNamespaces()
UnaryLogicOpType

Description
UnaryLogicOpType
UnaryLogicOpType

Format
R6Class object.

Value
Object of R6Class for modelling an UnaryLogicOpType

Super classes
ows4R::OGCAbstractObject -> ows4R::OGCExpression -> UnaryLogicOpType

Public fields
operations a list OGC expressions

Methods

Public methods:
• UnaryLogicOpType$new()
• UnaryLogicOpType$setExprVersion()
• UnaryLogicOpType$clone()

Method new(): Initializes a UnaryLogicOpType expression
Usage:
UnaryLogicOpType$new(..., element, namespacePrefix, exprVersion = "1.1.0")
Arguments:
... list of objects of class OGCExpression
element element
namespacePrefix namespacePrefix
exprVersion OGC expression version. Default is "1.1.0"

Method setExprVersion(): Sets expression version. The methods will control that expression versions are set for each of the operations specified in the expression.
Usage:
UnaryLogicOpType$setExprVersion(exprVersion)
Arguments:
**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

```r
UnaryLogicOpType$clone(deep = FALSE)
```

*Arguments:*

- `deep` Whether to make a deep clone.

**Note**

abstract super class of all the unary logical operation classes

---

### Description

**WCSCapabilities**

- WCSCapabilities

### Format

**R6Class** object.

### Value

Object of **R6Class** with methods for interfacing an OGC Web Coverage Service Get Capabilities document.

### Super classes

`ows4R::OGCAbstractObject` -> `ows4R::OWSCapabilities` -> **WCSCapabilities**

### Methods

**Public methods:**

- `WCSCapabilities$new()`
- `WCSCapabilities$getCoverageSummaries()`
- `WCSCapabilities$findCoverageSummaryById()`
- `WCSCapabilities$clone()`

**Method** `new()`: Initializes a **WCSCapabilities** object

*Usage:*

```r
WCSCapabilities$new(url, version, client = NULL, logger = NULL, ...)
```

*Arguments:*
url url
version version
client an object of class WCSClient
logger logger type NULL, "INFO" or "DEBUG"
... any other parameter to pass to OWSGetCapabilities service request

**Method** getCoverageSummaries(): Get coverage summaries

**Usage:**
WCSCapabilities$getCoverageSummaries()

**Returns:** a list of WCSCoverageSummary objects

**Method** findCoverageSummaryById(): Finds a coverage by name

**Usage:**
WCSCapabilities$findCoverageSummaryById(expr, exact = FALSE)

**Arguments:**
expr expr
exact exact matching? Default is TRUE

**Method** clone(): The objects of this class are cloneable with this method.

**Usage:**
WCSCapabilities$clone(deep = FALSE)

**Arguments:**
deep Whether to make a deep clone.

**Author(s)**
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

**Examples**

```r
## Not run:
WCSCapabilities$new("http://localhost:8080/geoserver/wcs", serviceVersion = "2.0.1")

## End(Not run)
```
**Description**

WCSClent

**Format**

*R6Class* object.

**Value**

Object of *R6Class* with methods for interfacing an OGC Web Coverage Service.

**Super classes**

ows4R::OGCAbstractObject -> ows4R::OWSClient -> WCSClent

**Methods**

**Public methods:**

- WCSClent$new()
- WCSClent$getCapabilities()
- WCSClent$reloadCapabilities()
- WCSClent$describeCoverage()
- WCSClent$getCoverage()
- WCSClent$clone()

**Method new():** This method is used to instantiate a WCSClent with the url of the OGC service. Authentication is supported using basic auth (using user/pwd arguments), bearer token (using token argument), or custom (using headers argument). By default, the logger argument will be set to NULL (no logger). This argument accepts two possible values: INFO: to print only ows4R logs, DEBUG: to print more verbose logs

**Usage:**

```r
WCSClent$new(
  url,
  serviceVersion = NULL,
  user = NULL,
  pwd = NULL,
  token = NULL,
  headers = c(),
  config = httr::config(),
  cas_url = NULL,
  logger = NULL
)
```
Arguments:
url url
serviceVersion WFS service version
user user
pwd password
token token
headers headers
config config
cas_url Central Authentication Service (CAS) URL
logger logger

Method `getCapabilities()`: Get WCS capabilities

Usage:
`WCSClient$getCapabilities()`

Returns: an object of class `WCSCapabilities`

Method `reloadCapabilities()`: Reloads WCS capabilities

Usage:
`WCSClient$reloadCapabilities()`

Method `describeCoverage()`: Describes coverage

Usage:
`WCSClient$describeCoverage(identifier)`

Arguments:
 identifier identifier

Returns: an object of class `WCSCoverageDescription`

Method `getCoverage()`: Get coverage

Usage:
`WCSClient$getCoverage(identifier, bbox = NULL, crs = NULL, time = NULL, format = NULL, rangesubset = NULL, gridbaseCRS = NULL, gridtype = NULL, gridCS = NULL, gridorigin = NULL, gridoffsets = NULL, method = "GET", filename = NULL, ...)`
Arguments:

- **identifier** Coverage identifier. Object of class character
- **bbox** bbox. Object of class matrix. Default is NULL. eg. OWSUtils$toBBOX(-180, 180, -90, 90)
- **crs** crs. Object of class character giving the CRS identifier (EPSG prefixed code, or URI/URN). Default is NULL.
- **time** time. Object of class character representing time instant/period. Default is NULL
- **format** format. Object of class character. Default will be GeoTIFF, coded differently depending on the WCS version.
- **rangesubset** rangesubset. Default is NULL
- **gridbaseCRS** grid base CRS. Default is NULL
- **gridtype** grid type. Default is NULL
- **gridCS** grid CS. Default is NULL
- **gridorigin** grid origin. Default is NULL
- **gridoffsets** grid offsets. Default is NULL
- **method** method to get coverage, either 'GET' or 'POST' (experimental - under development). Object of class character.
- **filename** filename. Object of class character. Optional filename to download the coverage...

Returns: an object of class SpatRaster from terra

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

`WCSClient$clone(deep = FALSE)`

Arguments:

- **deep** Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```r
## Not run:
wcs <- WCSClient$new("http://localhost:8080/geoserver/wcs", serviceVersion = "2.0.1")

## End(Not run)
```
WCSCoverage

Description

WCSCoverage

Format

R6Class object.

Value

Object of R6Class modelling a WCS coverage

Super class

ows4R::OGCAbstractObject -> WCSCoverage

Public fields

description

description

Methods

Public methods:

• WCSCoverage$new()
• WCSCoverage$getIdentifier()
• WCSCoverage$getTitle()
• WCSCoverage$getAbstract()
• WCSCoverage$getReference()
• WCSCoverage$getData()
• WCSCoverage$clone()

Method new(): Initializes an object of class WCSCoverage

Usage:

WCSCoverage$new(xmlObj, serviceVersion, owsVersion, logger = NULL)

Arguments:

xmlObj object of class XMLInternalNode-class
serviceVersion WCS service version
owsVersion OWS version
logger logger

Method getIdentifier(): Get identifier
Usage:
WCSCoverage$getIdentifier()

Returns: an object of class character

Method getTitle(): Get title
Usage:
WCSCoverage$getTitle()

Returns: an object of class character

Method getAbstract(): Get abstract
Usage:
WCSCoverage$getAbstract()

Returns: an object of class character

Method getReference(): Get reference
Usage:
WCSCoverage$getReference()

Returns: an object of class character

Method getData(): Get data
Usage:
WCSCoverage$getData(filename = NULL)

Arguments:
filename filename. Optional file name where to download the coverage

Returns: an object of class SpatRaster from terra

Method clone(): The objects of this class are cloneable with this method.
Usage:
WCSCoverage$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Note
Class used internally by ows4R.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
WCSCoverageDescription

Description
WCSCoverageDescription
WCSCoverageDescription

Format
R6Class object.

Value
Object of R6Class modelling a WCS coverage summary

Super classes
geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::GMLAbstractObject
-> geometa::GMLAbstractFeature -> geometa::GMLAbstractCoverage -> geometa::GMLCOVAbstractCoverage
-> WCSCoverageDescription

Public fields
CoverageId coverage ID
SupportedCRS supported CRS
SupportedFormat supported Format
Domain domain
Range range
ServiceParameters service parameters

Methods
Public methods:
• WCSCoverageDescription$new()
• WCSCoverageDescription$getId()
• WCSCoverageDescription$getSupportedCRS()
• WCSCoverageDescription$getSupportedFormats()
• WCSCoverageDescription$getDomain()
• WCSCoverageDescription$getRange()
• WCSCoverageDescription$clone()

Method new(): Initializes an object of class WCSCoverageDescription
Usage:
WCSCoverageDescription$new(xmlObj, serviceVersion, owsVersion, logger = NULL)

Arguments:
xmlObj an object of class XMLInternalNode-class to initialize from XML
serviceVersion service version
owsVersion OWS version
logger logger

Method getId(): getId
Usage:
WCSCoverageDescription$getId()
Returns: the coverage id, object of class character

Method getSupportedCRS(): getSupported CRS. Applies to WCS 1 coverage descriptions
Usage:
WCSCoverageDescription$getSupportedCRS()

Method getSupportedFormats(): get supported formats. Applies to WCS 1 coverage descriptions
Usage:
WCSCoverageDescription$getSupportedFormats()

Method getDomain(): get domain. Applies to WCS 1 coverage descriptions
Usage:
WCSCoverageDescription$getDomain()

Method getRange(): get range. Applies to WCS 1.0 coverage descriptions
Usage:
WCSCoverageDescription$getRange()

Method clone(): The objects of this class are cloneable with this method.
Usage:
WCSCoverageDescription$clone(deep = FALSE)
Arguments:
deep Whether to make a deep clone.

Note
Class used internally by ows4R.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
WCSCoverageDomain

Description

WCSCoverageDomain

Format

R6Class object.

Value

Object of R6Class modelling a WCS coverage domain

Super class

ows4R::OGCAbstractObject -> WCSCoverageDomain

Public fields

spatialDomain spatial domain
temporalDomain temporal domain

Methods

Public methods:

• WCSCoverageDomain$new()
• WCSCoverageDomain$getSpatialDomain()
• WCSCoverageDomain$getTemporalDomain()
• WCSCoverageDomain$clone()

Method new(): Initializes an object of class WCSCoverageDomain

Usage:
WCSCoverageDomain$new(xmlObj, serviceVersion, owsVersion, logger = NULL)

Arguments:
xmlObj an object of class XMLInternalNode-class to initialize from XML
serviceVersion service version
owsVersion OWS version
logger logger

Method getSpatialDomain(): Get spatial domain

Usage:
WCSCoverageDomain$getSpatialDomain()
**WCSCoverageFilenameHandler**

*Returns*: object of class `WCSCoverageSpatialDomain`

**Method** `getTemporalDomain()`: Get spatial domain

*Usage:*

```
WCSCoverageDomain$getTemporalDomain()
```

*Returns*: object of class `WCSCoverageTemporalDomain`

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

```
WCSCoverageDomain$clone(deep = FALSE)
```

*Arguments:*

deepe Whether to make a deep clone.

**Note**

Class used internally by ows4R.

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**WCSCoverageFilenameHandler**

**WCSCoverageFilenameHandler**

**Description**

`WCSCoverageFilenameHandler` provides a coverage filename handler for coverage download.

**Usage**

```
WCSCoverageFilenameHandler(identifier, time, elevation, bbox, format)
```

**Arguments**

- **identifier**: coverage identifier
- **time**: time
- **elevation**: elevation
- **bbox**: bbox
- **format**: format

**Value**

the filename to use for coverage download

**Author(s)**

Emmanuel Blondel, <emmanuel.blondel1@gmail.com>
WCSCoverageSpatialDomain

Description

WCSCoverageSpatialDomain
WCSCoverageSpatialDomain

Format

R6Class object.

Value

Object of R6Class modelling a WCS coverage spatial domain

Super class

ows4R::OGCAbstractObject -> WCSCoverageSpatialDomain

Public fields

envelopes envelopes. For WCS 1.0
BoundingBox bounding box. For WCS 1.1
 grids For WCS 1.0
 GridCRS grid CRS. For WCS 1.1

Methods

Public methods:

• WCSCoverageSpatialDomain$new()
• WCSCoverageSpatialDomain$getEnvelopes()
• WCSCoverageSpatialDomain$getBoundingBox()
• WCSCoverageSpatialDomain$getGrids()
• WCSCoverageSpatialDomain$getGridCRS()
• WCSCoverageSpatialDomain$clone()

Method new(): Initializes an object of class WCSCoverageDomain

Usage:

WCSCoverageSpatialDomain$new(xmlObj, serviceVersion, owsVersion, logger = NULL)

Arguments:

xmlObj an object of class XMLInternalNode-class to initialize from XML
serviceVersion service version
Method `getEnvelopes()`: Get envelopes. Method that applies to WCS 1.0 only

Usage:

```r
WCSCoverageSpatialDomain$getEnvelopes()
```

Returns: a list of objects of class `GMLEnvelope` or `GMLEnvelopeWithTimePeriod`

Method `getBoundingBox()`: Get bounding boxes. Method that applies to WCS 1.1 only

Usage:

```r
WCSCoverageSpatialDomain$getBoundingBox()
```

Returns: a list of objects of class `OWSBoundingBox`

Method `getGrids()`: Get grids. Method that applies to WCS 1.0 only

Usage:

```r
WCSCoverageSpatialDomain$getGrids()
```

Returns: a list of objects of class `GMLGrid` or `GMLRectifiedGrid`

Method `getGridCRS()`: Get Grid CRS. Method that applies to WCS 1.1 only

Usage:

```r
WCSCoverageSpatialDomain$getGridCRS()
```

Returns: a list of objects of class `WCSGridCRS`

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```r
WCSCoverageSpatialDomain$clone(deep = FALSE)
```

Arguments:

- `deep` Whether to make a deep clone.

**Note**

Class used internally by ows4R.

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Description

WCSCoverageSummary

Format

R6Class object.

Value

Object of R6Class modelling a WCS coverage summary

Super class

ows4R::OGCAbstractObject -> WCSCoverageSummary

Public fields

CoverageId coverage id
CoverageSubtype coverage subtype
CoverageSubtypeParent coverage subtype parent
WGS84BoundingBox WGS84 bounding box
BoundingBox bounding box

Methods

Public methods:

- WCSCoverageSummary$new()
- WCSCoverageSummary$getId()
- WCSCoverageSummary$getSubtype()
- WCSCoverageSummary$getSubtypeParent()
- WCSCoverageSummary$getWGS84BoundingBox()
- WCSCoverageSummary$getBoundingBox()
- WCSCoverageSummary$getDescription()
- WCSCoverageSummary$getDimensions()
- WCSCoverageSummary$getCoverage()
- WCSCoverageSummary$getCoverageStack()
- WCSCoverageSummary$clone()

Method new(): Initializes a WCSCoverageSummary object
Usage:
WCSCoverageSummary$new(
  xmlObj,
  capabilities,
  serviceVersion,
  owsVersion,
  logger = NULL
)

Arguments:
xmlObj object of class XMLInternalNode-class from XML
capabilities object of class WCSCapabilities
serviceVersion WCS service version
owsVersion version
logger logger type NULL, "INFO" or "DEBUG"

Method getId(): Get coverage ID
Usage:
WCSCoverageSummary$getId()
Returns: an object of class character

Method getSubtype(): Get sub type
Usage:
WCSCoverageSummary$getSubtype()
Returns: an object of class character

Method getSubtypeParent(): Get sub type parent
Usage:
WCSCoverageSummary$getSubtypeParent()
Returns: an object of class character

Method getWGS84BoundingBox(): Get bounding box
Usage:
WCSCoverageSummary$getWGS84BoundingBox()
Returns: an object of class OWSWGS84BoundingBox

Method getBoundingBox(): Get WGS84 bounding box
Usage:
WCSCoverageSummary$getBoundingBox()
Returns: an object of class OWSBoundingBox

Method getDescription(): Get description
Usage:
WCSCoverageSummary$getDescription()
Returns: an object of class `WCSCoverageDescription`

**Method** `getDimensions()`: Get dimensions

*Usage:*  
`WCSCoverageSummary$getDimensions()`  
*Returns:* the list of dimensions

**Method** `getCoverage()`: Get coverage data

*Usage:*  
`WCSCoverageSummary$getCoverage(bbox = NULL, crs = NULL, time = NULL, elevation = NULL, format = NULL, rangesubset = NULL, gridbaseCRS = NULL, gridtype = NULL, gridCS = NULL, gridorigin = NULL, gridoffsets = NULL, method = "GET", filename = NULL, ...)`

*Arguments:*  
- `bbox` bbox. Object of class `matrix`. Default is `NULL`. eg. `OWSUtils$toBBOX(-180,180,-90,90)`  
- `crs` crs. Object of class `character` giving the CRS identifier (EPSG prefixed code, or URI/URN). Default is `NULL`.  
- `time` time. Object of class `character` representing time instant/period. Default is `NULL`.  
- `elevation` elevation. Object of class `character` or numeric. Default is `NULL`.  
- `format` format. Object of class `character` Default will be GeoTIFF, coded differently depending on the WCS version.  
- `rangesubset` rangesubset. Default is `NULL`.  
- `gridbaseCRS` grid base CRS. Default is `NULL`.  
- `gridtype` grid type. Default is `NULL`.  
- `gridCS` grid CS. Default is `NULL`.  
- `gridorigin` grid origin. Default is `NULL`.  
- `gridoffsets` grid offsets. Default is `NULL`.  
- `method` method to get coverage, either 'GET' or 'POST' (experimental - under development). Object of class character.  
- `filename` filename. Object of class character. Optional filename to download the coverage ... any other argument to `WCSGetCoverage`  

*Returns:* an object of class `SpatRaster` from `terra`
Method `getCoverageStack()`: Get a spatio-temporal coverage data cubes as coverage stack

Usage:
```r
WCSCoverageSummary$getCoverageStack(
    time = NULL,
    elevation = NULL,
    bbox = NULL,
    filename_handler = NULL,
    ...
)
```

Arguments:
- `time` time
elevation elevation
bbox bbox
filename_handler Optional filename handling function with arguments 'identifier', 'time', 'elevation', 'bbox', 'format' See `WCSCoverageFilenameHandler` as generic filename handler that can be used.
... any other parameter to pass to `getCoverage`

Returns: an object of class `stack` from `raster`

Method `clone()`: The objects of this class are cloneable with this method.

Usage:
```r
WCSCoverageSummary$clone(deep = FALSE)
```

Arguments:
- `deep` Whether to make a deep clone.

Note
Class used internally by ows4R.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Description

Format

`R6Class` object.
Value

Object of R6Class modelling a WCS coverage temporal domain

Super class

ows4R::OGCAbstractObject -> WCSCoverageTemporalDomain

Public fields

  instants
  periods

Methods

Public methods:

- WCSCoverageTemporalDomain$new()
- WCSCoverageTemporalDomain$getInstants()
- WCSCoverageTemporalDomain$getPeriods()
- WCSCoverageTemporalDomain$clone()

Method $new()$: Initializes an object of class WCSCoverageTemporalDomain

Usage:

WCSCoverageTemporalDomain$new(xmlObj, serviceVersion, owsVersion, logger = NULL)

Arguments:

xmlObj an object of class XMLInternalNode-class to initialize from XML
serviceVersion service version
owsVersion OWS version
logger logger

Method $getInstants()$: Get time instants

Usage:

WCSCoverageTemporalDomain$getInstants()

Returns: a list of objects of class POSIXct

Method $getPeriods()$: Get time periods

Usage:

WCSCoverageTemporalDomain$getPeriods()

Returns: a list of objects of class GMLTimePeriod

Method $clone()$: The objects of this class are cloneable with this method.
Usage:
WCSCoverageTemporalDomain$clone(deep = FALSE)

Arguments:
depth Whether to make a deep clone.

Note
Class used internally by ows4R.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Description
WCSDescribeCoverage
WCSDescribeCoverage

Format
R6Class object.

Value
Object of R6Class for modelling a WCS DescribeCoverage request

Super classes
ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> WCSDescribeCoverage

Methods
Public methods:
• WCSDescribeCoverage$new()
• WCSDescribeCoverage$clone()

Method new(): Initializes a WCSDescribeCoverage service request

Usage:
WCSDescribeCoverage$new(
  capabilities,
  op,
  url,
  serviceVersion,
  coverageId,
  user = NULL,
  pwd = NULL,
  token = NULL,
  headers = c(),
  config = httr::config(),
  logger = NULL,
  ...
)

Arguments:
capabilities an object of class WCSCapabilities
op object of class OWSOperation as retrieved from capabilities
url url
serviceVersion serviceVersion
coverageId coverage ID
user user
pwd password
token token
headers headers
config config
logger logger
... any parameter to pass to the service request

Method clone(): The objects of this class are cloneable with this method.

Usage:
WCSDescribeCoverage$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Note
Class used internally by ows4R to trigger a WCS DescribeCoverage request

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
**WCSGetCoverage**

**Description**
WCSGetCoverage
WCSGetCoverage

**Format**

*R6Class* object.

**Value**

Object of *R6Class* for modelling a WCS GetCoverage request

**Super classes**

`ows4R::OGCAbstractObject` -> `ows4R::OWSHttpRequest` -> `WCSGetCoverage`

**Public fields**

- `CoverageId` coverage identifier
- `format` coverage format

**Methods**

**Public methods:**

- `WCSGetCoverage$new()`
- `WCSGetCoverage$clone()`

**Method** `new()`: Initializes a *WCSGetCoverage* service request

**Usage:**

```r
WCSGetCoverage$new(
capabilities, 
op, 
url, 
serviceVersion, 
coverage, 
envelope = NULL, 
crs = NULL, 
time = NULL, 
elevation = NULL, 
format = NULL, 
rangesubset = NULL, 
gridbaseCRS = NULL,
```
gridtype = NULL,
gridCS = NULL,
gridorigin = NULL,
gridoffsets = NULL,
user = NULL,
pwd = NULL,
token = NULL,
headers = c(),
config = httr::config(),
method = "GET",
logger = NULL,
...
)

Arguments:
capabilities an object of class WCSCapabilities
op object of class OWSOperation as retrieved from capabilities
url url
serviceVersion serviceVersion
coverage coverage, object of class WCSCoverageSummary
envelope envelope
crs crs
time time
elevation elevation
format format
rangesubset range subset
gridbaseCRS grid base CRS
gridtype grid type
gridCS grid CS
gridorigin grid origin
gridoffsets grid offsets
user user
pwd password
token token
headers headers
config config
method method
logger logger
... any parameter to pass to the service request

Method clone(): The objects of this class are cloneable with this method.

Usage:
WCSGetCoverage$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.
Note

Class used internally by ows4R to trigger a WCS GetCoverage request

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**Description**

WCSGridCRS  
WCSGridCRS

**Format**

*R6Class* object.

**Value**

Object of *R6Class* modelling a WCS grid CRS

**Super class**

*ows4R::OGCAbstractObject* -> WCSGridCRS

**Public fields**

- GridBaseCRS  grid base CRS
- GridType  grid type
- GridOrigin  grid origin
- GridOffsets  grid offsets
- GridCS  grid CS

**Methods**

**Public methods:**

- `WCSGridCRS$new()`  
- `WCSGridCRS$clone()`

**Method** `new()`: Initializes an object of class *WCSGridCRS*

*Usage:*

`WCSGridCRS$new(xmlObj, serviceVersion, owsVersion, logger = NULL)`

*Arguments:*
xmlObj an object of class \texttt{XMLInternalNode-class} to initialize from XML

\texttt{serviceVersion} service version

\texttt{owsVersion} OWS version

\texttt{logger} logger

\textbf{Method} \texttt{clone()}: The objects of this class are cloneable with this method.

\texttt{Usage: WCSGridCRS$clone(deep = FALSE)}

\texttt{Arguments:}

deepl Whether to make a deep clone.

\textbf{Note}

Class used internally by \texttt{ows4R}.

\textbf{Author(s)}

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

\textbf{Description}

WCSGSElevationDomain

\textbf{Format}

\texttt{R6Class} object.

\textbf{Value}

Object of \texttt{R6Class} modelling a WCS geoserver elevation domain object

\textbf{Super classes}

\texttt{geometa::geometaLogger} -> \texttt{geometa::ISOAbstractObject} -> WCSGSElevationDomain

\textbf{Public fields}

\texttt{SingleValue} single values
WCSGSTimeDomain

Methods

Public methods:

• WCSGSElevationDomain$decode()
• WCSGSElevationDomain$clone()

Method decode(): Decodes from XML

Usage:
WCSGSElevationDomain$decode(xml)

Arguments:
xml object of class XMLInternalNode-class from XML

Method clone(): The objects of this class are cloneable with this method.

Usage:
WCSGSElevationDomain$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

WCSGSTimeDomain       WCSGSTimeDomain

Description

WCSGSTimeDomain
WCSGSTimeDomain

Format

R6Class object.

Value

Object of R6Class modelling a WCS geoserver time domain object

Super classes

gometac::gometalogger -> gometac::ISOAbstractObject -> WCSGSTimeDomain
Public fields

TimeInstant time instants

Methods

Public methods:

- WCSGSTimeDomain$decode()
- WCSGSTimeDomain$clone()

Method decode(): Decodes from XML

Usage:

WCSGSTimeDomain$decode(xml)

Arguments:

xml object of class XMLInternalNode-class from XML

Method clone(): The objects of this class are cloneable with this method.

Usage:

WCSGSTimeDomain$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

WFSCapabilities

Description

WFSCapabilities

Format

R6Class object.

Value

Object of R6Class with methods for interfacing an OGC Web Feature Service Get Capabilities document.
**WFSCapabilities**

**Super classes**

ows4R::OGCAbstractObject -> ows4R::OWSCapabilities -> WFSCapabilities

**Methods**

**Public methods:**

- WFSCapabilities$new()
- WFSCapabilities$getFeatureTypes()
- WFSCapabilities$findFeatureTypeByName()
- WFSCapabilities$clone()

**Method new():** Initializes a WFSCapabilities object

*Usage:*

WFSCapabilities$new(url, version, logger = NULL, ...)

*Arguments:*

- **url**
- **version**
- **logger** logger type NULL, "INFO" or "DEBUG"
- **...** any other parameter to pass to OWSGetCapabilities service request

**Method getFeatureTypes():** List the feature types available. If pretty is TRUE, the output will be an object of class data.frame

*Usage:*

WFSCapabilities$getFeatureTypes(pretty = FALSE)

*Arguments:*

- **pretty** whether the output should be summarized as data.frame

*Returns:* a list of WFSFeatureType or a data.frame

**Method findFeatureTypeByName():** Finds a feature type by name

*Usage:*

WFSCapabilities$findFeatureTypeByName(expr, exact = TRUE)

*Arguments:*

- **expr**
- **exact** exact matching? Default is TRUE

**Method clone():** The objects of this class are cloneable with this method.

*Usage:*

WFSCapabilities$clone(deep = FALSE)

*Arguments:*

- **deep** Whether to make a deep clone.

**Note**

Class used to read a WFSCapabilities document. The use of WFSClient is recommended instead to benefit from the full set of capabilities associated to a WFS server.
WFSClent

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Description

WFSClent

WFSClent

Format

R6Class object.

Value

Object of R6Class with methods for interfacing an OGC Web Feature Service.

Super classes

ows4R::OGCAbstractObject -> ows4R::OWSClient -> WFSClent

Methods

Public methods:

- WFSClent$new()
- WFSClent$getCapabilities()
- WFSClent$reloadCapabilities()
- WFSClent$describeFeatureType()
- WFSClent$getFeatures()
- WFSClent$getFeatureTypes()
- WFSClent$clone()

Method new(): This method is used to instantiate a WFSClent with the url of the OGC service. Authentication is supported using basic auth (using user/pwd arguments), bearer token (using token argument), or custom (using headers argument). By default, the logger argument will be set to NULL (no logger). This argument accepts two possible values: INFO: to print only ows4R logs, DEBUG: to print more verbose logs

Usage:

WFSClent$new(
    url,
    serviceVersion = NULL,
    user = NULL,
    pwd = NULL,
)
token = NULL,
headers = c(),
config = httr::config(),
cas_url = NULL,
logger = NULL
)

Arguments:
url  url
serviceVersion  WFS service version
user  user
pwd  password
token  token
headers  headers
config  config
cas_url  Central Authentication Service (CAS) URL
logger  logger

Method getCapabilities(): Get WFS capabilities
  Usage:
  WFSClient$getCapabilities()
  Returns: an object of class WFSCapabilities

Method reloadCapabilities(): Reloads WFS capabilities
  Usage:
  WFSClient$reloadCapabilities()

Method describeFeatureType(): Describes a feature type
  Usage:
  WFSClient$describeFeatureType(typeName)
  Arguments:
  typeName  the name of the feature type
  Returns: a list of WFSGlobalFeatureTypeElement

Method getFeatures(): Get features
  Usage:
  WFSClient$getFeatures(typeName, ...)
  Arguments:
  typeName  the name of the feature type
  ...  any other parameter to pass to the WFSGetFeature request
  Returns: features as object of class sf

Method getFeatureTypes(): List the feature types available. If pretty is TRUE, the output will be an object of class data.frame
WFSDescribeFeatureType

Usage:
WFSClient$getFeatureTypes(pretty = FALSE)

Arguments:
pretty whether the output should be summarized as data.frame

Returns: a list of WFSFeatureType or a data.frame

Method clone(): The objects of this class are cloneable with this method.

Usage:
WFSClient$clone(deep = FALSE)

Arguments:
dee Whether to make a deep clone.

Author(s)
Emmanuel Blondel <emmanuel.blondell1@gmail.com>

Examples

```r
## Not run:
# example based on a WFS endpoint responding at http://localhost:8080/geoserver/wfs
wfs <- WFSClient$new("http://localhost:8080/geoserver/wfs", serviceVersion = "1.1.1")

gét capabilities
caps <- wfs$getCapabilities()

# find feature type
ft <- caps$findFeatureTypeByName("mylayer")
if(length(ft)>0){
  data <- ft$getFeatures()
  data_with_filter <- ft$getFeatures(cql_filter = "somefilter")
}

# Advanced examples at https://github.com/eblondel/ows4R/wiki#wfs

## End(Not run)
```
**Format**

R6Class object.

**Value**

Object of R6Class for modelling a WFS DescribeFeatureType request

**Super classes**

ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> WFSDescribeFeatureType

**Methods**

**Public methods:**

- WFSDescribeFeatureType$new()
- WFSDescribeFeatureType$clone()

**Method** `new()`: Initializes a WFSDescribeFeatureType service request

**Usage:**

WFSDescribeFeatureType$new(
  capabilities,
  op,
  url,
  version,
  typeName,
  user = NULL,
  pwd = NULL,
  token = NULL,
  headers = c(),
  config = httr::config(),
  logger = NULL,
  ...
)

**Arguments:**

capabilities an object of class WFSCapabilities
op object of class OWSOperation as retrieved from capabilities
url url
version service version
typeName typeName
user user
pwd pwd
token token
headers headers
config config
logger logger
... any parameter to pass to the service request
Method `clone()`: The objects of this class are cloneable with this method.

Usage:
`WFSDescribeFeatureType$clone(deep = FALSE)`

Arguments:
deep Whether to make a deep clone.

Note
Abstract class used by `ows4R` to trigger a WFS DescribeFeatureType request

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

| WFSFeatureType | WFSFeatureType |

Description
WFSFeatureType
WFSFeatureType

Format
`R6Class` object.

Value
Object of `R6Class` modelling a WFS feature type

Super class
`ows4R::OGCAbstractObject` -> WFSFeatureType

Public fields
description description
features features
Methods

Public methods:

• WFSFeatureType$new()
• WFSFeatureType$getName()
• WFSFeatureType$getTitle()
• WFSFeatureType$getAbstract()
• WFSFeatureType$getKeywords()
• WFSFeatureType$getDefaultCRS()
• WFSFeatureType$getBoundingBox()
• WFSFeatureType$getDescription()
• WFSFeatureType$hasGeometry()
• WFSFeatureType$getGeometryType()
• WFSFeatureType$getFeaturesCRS()
• WFSFeatureType$getFeatures()
• WFSFeatureType$clone()

Method new(): Initialize an object of class WFSFeatureType

Usage:
WFSFeatureType$new(xmlObj, capabilities, version, logger = NULL)

Arguments:
xmlObj an object of class XMLInternalNode-class to initialize from XML
capabilities object of class WFSCapabilities
version service version
logger logger

Method getName(): Get feature type name

Usage:
WFSFeatureType$getName()

Arguments:
object of class character

Method getTitle(): Get feature type title

Usage:
WFSFeatureType$getTitle()

Arguments:
object of class character

Method getAbstract(): Get feature type abstract

Usage:
WFSFeatureType$getAbstract()

Arguments:
object of class character
Method getKeywords(): Get feature type keywords

Usage:
WFSFeatureType$getKeywords()

Arguments:
object of class character

Method getDefaultCRS(): Get feature type default CRS

Usage:
WFSFeatureType$getDefaultCRS()

Arguments:
object of class character

Method getBoundingBox(): Get feature type bounding box

Usage:
WFSFeatureType$getBoundingBox()

Arguments:
object of class matrix

Method getDescription(): Describes a feature type

Usage:
WFSFeatureType$getDescription(pretty = FALSE)

Arguments:
pretty pretty whether to return a prettified data.frame. Default is FALSE

Returns: a list of WFSFeatureTypeElement or data.frame

Method hasGeometry(): Indicates with feature type has a geometry

Usage:
WFSFeatureType$hasGeometry()

Returns: object of class logical

Method getGeometryType(): Get geometry type

Usage:
WFSFeatureType$getGeometryType()

Returns: object of class character representing the geometry type

Method getFeaturesCRS(): Inherits features CRS

Usage:
WFSFeatureType$getFeaturesCRS(obj)

Arguments:
obj features object

Returns: object of class integer
Method getFeatures(): Get features

Usage:
WFSFeatureType$getFeatures(
  ..., 
  outputFormat = NULL, 
  paging = FALSE, 
  paging_length = 1000, 
  parallel = FALSE, 
  parallel_handler = NULL, 
  cl = NULL 
)

Arguments:
... any other parameter to pass to the WFSGetFeature request 
outputFormat output format 
paging paging. Default is FALSE 
paging_length number of features to request per page. Default is 1000 
parallel whether to get features using parallel multicore strategy. Default is FALSE 
parallel_handler Handler function to parallelize the code. eg mclapply 
cl optional cluster object for parallel cluster approaches using eg. parallel::makeCluster 
typeName the name of the feature type 

Returns: features as object of class sf

Method clone(): The objects of this class are cloneable with this method.

Usage:
WFSFeatureType$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Note
Class used internally by ows4R to trigger a WFS DescribeFeatureType request

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Format

*R6Class* object.

Value

Object of *R6Class* modelling a WFS feature type element

Methods

**Public methods:**

- `WFSFeatureTypeElement$new()`
- `WFSFeatureTypeElement$getMinOccurs()`
- `WFSFeatureTypeElement$getMaxOccurs()`
- `WFSFeatureTypeElement$isNillable()`
- `WFSFeatureTypeElement$getName()`
- `WFSFeatureTypeElement$getType()`
- `WFSFeatureTypeElement$isGeometry()`
- `WFSFeatureTypeElement$clone()`

**Method** `new()`: Initializes a `WFSFeatureTypeElement`

*Usage:*

```r
WFSFeatureTypeElement$new(xmlObj)
```

*Arguments:*

`xmlObj` object of class `XMLInternalNode-class` from `XML`

**Method** `getMinOccurs()`: get min occurs

*Usage:*

```r
WFSFeatureTypeElement$getMinOccurs()
```

*Returns: an object of class character*

**Method** `getMaxOccurs()`: get max occurs

*Usage:*

```r
WFSFeatureTypeElement$getMaxOccurs()
```

*Returns: an object of class character*

**Method** `isNillable()`: get if nillable

*Usage:*

```r
WFSFeatureTypeElement$isNillable()
```

*Returns: an object of class logical*

**Method** `getName()`: get name

*Usage:*

```r
WFSFeatureTypeElement$getName()
```

*Returns: an object of class character*
Method getType(): get type
Usage:
WFSFeatureTypeElement$getType()
Returns: an object of class character

Method isGeometry(): Is geometry
Usage:
WFSFeatureTypeElement$isGeometry()
Arguments:
return object of class logical

Method clone(): The objects of this class are cloneable with this method.
Usage:
WFSFeatureTypeElement$clone(deep = FALSE)
Arguments:
deep Whether to make a deep clone.

Note
Abstract class used by ows4R

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

WFSGetFeature

Description
WFSGetFeature
WFSGetFeature

Format
R6Class object.

Value
Object of R6Class for modelling a WFS GetFeature request

Super classes
ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> WFSGetFeature
Methods

Public methods:

• WFSGetFeature$new()
• WFSGetFeature$clone()

Method new(): Initializes a WFSGetFeature service request

Usage:
WFSGetFeature$new(
    capabilities,
    op,
    url,
    version,
    typeName,
    outputFormat = NULL,
    user = NULL,
    pwd = NULL,
    token = NULL,
    headers = c(),
    config = httr::config(),
    logger = NULL,
    ...
)

Arguments:
capabilities an object of class WFSCapabilities
op object of class OWSOperation as retrieved from capabilities
url url
version service version
typeName typeName
outputFormat output format
user user
pwd password
token token
headers headers
config config
logger logger
... any parameter to pass to the service request

Method clone(): The objects of this class are cloneable with this method.

Usage:
WFSGetFeature$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.
**Note**

Class used internally by ows4R to trigger a WFS GetFeature request

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**WMSCapabilities**

**Description**

WMSCapabilities

**Format**

*R6Class* object.

**Value**

Object of *R6Class* with methods for interfacing an OGC Web Map Service Get Capabilities document.

**Super classes**

ows4R::OGCAbstractObject -> ows4R::OWSCapabilities -> WMSCapabilities

**Methods**

**Public methods:**

- WMSCapabilities$new()
- WMSCapabilities$getRequests()
- WMSCapabilities$getRequestNames()
- WMSCapabilities$getLayers()
- WMSCapabilities$findLayerByName()
- WMSCapabilities$clone()

**Method** `new()`: Initializes a WMSCapabilities object

**Usage:**

`WMSCapabilities$new(url, version, logger = NULL, ...)`

**Arguments:**

- `url`  url
- `version`  version
- `logger`  logger type NULL, "INFO" or "DEBUG"
any other parameter to pass to OWSGetCapabilities service request

Method `getRequests()`: List the requests available. If pretty is TRUE, the output will be an object of class `data.frame`

Usage:
`WMSCapabilities$getRequests(pretty = FALSE)`

Arguments:
pretty pretty

Returns: a list of `OWSRequest` available, or a data.frame

Method `getRequestNames()`: List the names of requests available.

Usage:
`WMSCapabilities$getRequestNames()`

Returns: object of class character

Method `getLayers()`: List the layers available. If pretty is TRUE, the output will be an object of class `data.frame`

Usage:
`WMSCapabilities$getLayers(pretty = FALSE)`

Arguments:
pretty pretty

Returns: a list of `WMSLayer` available, or a data.frame

Method `findLayerByName()`: Finds a layer by name

Usage:
`WMSCapabilities$findLayerByName(expr, exact = TRUE)`

Arguments:
expr expr
exact exact matching? Default is TRUE

Method `clone()`: The objects of this class are cloneable with this method.

Usage:
`WMSCapabilities$clone(deep = FALSE)`

Arguments:
deep Whether to make a deep clone.

Note

Class used to read a `WMSCapabilities` document. The use of `WMSClient` is recommended instead to benefit from the full set of capabilities associated to a WMS server.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Description

WMSClient

Format

R6Class object.

Value

Object of R6Class with methods for interfacing an OGC Web Map Service.

Super classes

ows4R::OGCAbstractObject -> ows4R::OWSClient -> WMSClient

Methods

Public methods:

• WMSClient$new()
• WMSClient$getCapabilities()
• WMSClient$reloadCapabilities()
• WMSClient$getLayers()
• WMSClient$getMap()
• WMSClient$getFeatureInfo()
• WMSClient$getLegendGraphic()
• WMSClient$clone()

Method new(): This method is used to instantiate a WMSClient with the url of the OGC service. Authentication is supported using basic auth (using user/pwd arguments), bearer token (using token argument), or custom (using headers argument). By default, the logger argument will be set to NULL (no logger). This argument accepts two possible values: INFO: to print only ows4R logs, DEBUG: to print more verbose logs

Usage:

WMSClient$new(
  url,  
serviceVersion = NULL,  
user = NULL,  
pwd = NULL,  
token = NULL,  
token = NULL,  
headers = c(),
config = httr::config(),
cas_url = NULL,
logger = NULL
)

Arguments:
url url
serviceVersion WFS service version
user user
pwd password
token token
headers headers
config config
cas_url Central Authentication Service (CAS) URL
logger logger

Method getCapabilities(): Get WMS capabilities
  Usage:
  WMSClient$getCapabilities()
  Returns: an object of class WMSCapabilities

Method reloadCapabilities(): Reloads WFS capabilities
  Usage:
  WMSClient$reloadCapabilities()

Method getLayers(): List the layers available. If pretty is TRUE, the output will be an object of class data.frame
  Usage:
  WMSClient$getLayers(pretty = FALSE)
  Arguments:
  pretty pretty
  Returns: a list of WMSLayer available, or a data.frame

Method getMap(): Get map. NOT YET IMPLEMENTED
  Usage:
  WMSClient$getMap()

Method getFeatureInfo(): Get feature info
  Usage:
  WMSClient$getFeatureInfo(
    layer,
    srs = NULL,
    styles = NULL,
    feature_count = 1,
    x,
WMSClient

y, width, height, bbox, info_format = "application/vnd.ogc.gml",
... )

Arguments:
layer layer name
srs srs
styles styles
feature_count feature count. Default is 1
x x
y y
width width
height height
bbox bbox
info_format info format. Default is "application/vnd.ogc.gml"
... any other parameter to pass to a WMSGetFeatureInfo request
Returns: an object of class sf given the feature(s)

Method getLegendGraphic(): Get legend graphic. NOT YET IMPLEMENTED
Usage:
WMSClient$getLegendGraphic()

Method clone(): The objects of this class are cloneable with this method.
Usage:
WMSClient$clone(deep = FALSE)
Arguments:
deep Whether to make a deep clone.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

## Not run:
# example based on a WMS endpoint responding at http://localhost:8080/geoserver/wms
wms <- WMSClient$new("http://localhost:8080/geoserver/wms", serviceVersion = "1.1.1")

#get capabilities
caps <- wms$getCapabilities()

#get feature info
Description

WMSGetFeatureInfo

Format

R6Class object.

Value

Object of R6Class for modelling a WMS GetFeatureInfo request

Super classes

ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> WMSGetFeatureInfo

Methods

Public methods:

• WMSGetFeatureInfo$new()
• WMSGetFeatureInfo$clone()

Method new(): Initializes a WMSGetFeatureInfo service request

Usage:

WMSGetFeatureInfo$new(
  capabilities,
  op,
  url,
  version,
  layers,
  srs,
  styles,
  feature_count = 1,
  x,
  y,
  width,
  height,
  bbox,
)
WMSGetFeatureInfo

```r
info_format = "text/xml",
user = NULL,
pwd = NULL,
token = NULL,
headers = c(),
config = httr::config(),
logger = NULL,
...
)
```

**Arguments:**
capabilities an object of class `WMSCapabilities`
op object of class `OWSOperation` as retrieved from capabilities
url url
version version
layers layers
srs srs
styles styles
feature_count feature count
x x
y y
width width
height height
bbox bbox
info_format info format
user user
pwd pwd
token token
headers headers
config config
logger logger
... any parameter to pass to the service request

**Method clone():** The objects of this class are cloneable with this method.

**Usage:**

```r
WMSGetFeatureInfo$clone(deep = FALSE)
```

**Arguments:**
deepl Whether to make a deep clone.

**Note**

Class used internally by `ows4R` to trigger a WMS GetFeatureInfo request

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
WMSLayer

Description

WMSLayer
WMSLayer

Format

R6Class object.

Value

Object of R6Class modelling a WMS layer

Super class

ows4R::OGCAbstractObject -> WMSLayer

Public fields

description
description
features
features

Methods

Public methods:

• WMSLayer$new()
• WMSLayer$getName()
• WMSLayer$getTitle()
• WMSLayer$getAbstract()
• WMSLayer$getKeywords()
• WMSLayer$getDefaultCRS()
• WMSLayer$getBoundingBox()
• WMSLayer$getBoundingBoxSRS()
• WMSLayer$getBoundingBoxCRS()
• WMSLayer$getStyles()
• WMSLayer$getStylenames()
• WMSLayer$getDimensions()
• WMSLayer$getTimeDimension()
• WMSLayer$getElevationDimension()
• WMSLayer$getFeatureInfo()
• WMSLayer$clone()
Method `new()`: Initializes an object of class `WMSLayer`

**Usage:**
```r
WMSLayer$new(xmlObj, capabilities, version, logger = NULL)
```

**Arguments:**
- `xmlObj`: an object of class `XMLInternalNode-class` to initialize from XML
- `capabilities`: object of class `WMSCapabilities`
- `version`: service version
- `logger`: logger

**Method `getName()`**: Get layer name

**Usage:**
```r
WMSLayer$getName()
```

**Returns:** object of class `character`

**Method `getTitle()`**: Get layer title

**Usage:**
```r
WMSLayer$getTitle()
```

**Returns:** object of class `character`

**Method `getAbstract()`**: Get layer abstract

**Usage:**
```r
WMSLayer$getAbstract()
```

**Returns:** object of class `character`

**Method `getKeywords()`**: Get layer keywords

**Usage:**
```r
WMSLayer$getKeywords()
```

**Returns:** object of class `character`

**Method `getDefaultCRS()`**: Get layer default CRS

**Usage:**
```r
WMSLayer$getDefaultCRS()
```

**Returns:** object of class `character`

**Method `getBoundingBox()`**: Get layer bounding box

**Usage:**
```r
WMSLayer$getBoundingBox()
```

**Returns:** object of class `matrix`

**Method `getBoundingBoxSRS()`**: Get layer bounding box SRS

**Usage:**
```r
WMSLayer$getBoundingBoxSRS()
```
**Method** `getBoundingBoxCRS()`: Get layer bounding box CRS

**Usage:**

`WMSLayer$getBoundingBoxCRS()`

**Returns:** object of class `character`

**Method** `getStyles()`: Get layer styles

**Usage:**

`WMSLayer$getStyles()`

**Returns:** an object of class `list`

**Method** `getStylenames()`: Get layer style names

**Usage:**

`WMSLayer$getStylenames()`

**Returns:** list of object of class `character`

**Method** `getDimensions()`: Get layer dimensions

**Usage:**

`WMSLayer$getDimensions(time_format = "character")`

**Arguments:**

time_format time format. Default is `character`

**Returns:** a list including default value and listed possible values

**Method** `getTimeDimension()`: Get layer TIME dimensions

**Usage:**

`WMSLayer$getTimeDimension(time_format = "character")`

**Arguments:**

time_format time format. Default is `character`

**Returns:** a list including default value and listed possible values

**Method** `getElevationDimension()`: Get layer ELEVATION dimensions

**Usage:**

`WMSLayer$getElevationDimension()`

**Returns:** a list including default value and listed possible values

**Method** `getFeatureInfo()`: Get feature info

**Usage:**
WMSLayer$getFeatureInfo(
  srs = NULL,
  styles = NULL,
  feature_count = 1,
  x,
  y,
  width,
  height,
  bbox,
  info_format = "text/xml",
  ...
)

*Arguments:*
srs srs
styles styles
feature_count feature count. Default is 1
x x
y y
width width
height height
bbox bbox
info_format info format. Default is "text/xml"
... any other parameter to pass to a WMSGetFeatureInfo request

*Returns:* an object of class sf given the feature(s)

**Method** clone(): The objects of this class are cloneable with this method.

*Usage:*
WMSLayer$clone(deep = FALSE)

*Arguments:*
deep Whether to make a deep clone.

**Note**
Abstract class used by ows4R

**Author(s)**
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Description

WPSBoundingBoxData
WPSBoundingBoxData

Format

R6Class object.

Value

Object of R6Class for modelling an OGC WPS BoundingBox data

Super classes

ows4R::OGCAbstractObject -> ows4R::OWSBoundingBox -> WPSBoundingBoxData

Methods

Public methods:
• WPSBoundingBoxData$new()
• WPSBoundingBoxData$clone()

Method new(): Initializes an object of class WPSBoundingBoxData

Usage:
WPSBoundingBoxData$new(
  xml = NULL,
  owsVersion,
  serviceVersion = "1.0.0",
  logger = NULL,
  ...
)

Arguments:
xml an object of class XMLInternalNode-class to initialize from XML
owsVersion OWS version
serviceVersion WPS service version
logger logger
... any other parameter

Method clone(): The objects of this class are cloneable with this method.

Usage:
WPSBoundingBoxData$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.
**Description**

WPSCapabilities

**Format**

R6Class object.

**Value**

Object of R6Class with methods for interfacing an OGC Web Processing Service (WPS) Get Capabilities document.

**Super classes**

ows4R::OGCAbstractObject -> ows4R::OWSCapabilities -> WPSCapabilities

**Methods**

Public methods:

- WPSCapabilities$new()
- WPSCapabilities$getProcesses()
- WPSCapabilities$describeProcess()
- WPSCapabilities$execute()
- WPSCapabilities$clone()

**Method new:** Initializes a WPSCapabilities object

*Usage:*

WPSCapabilities$new(url, version, client = NULL, logger = NULL, ...)

*Arguments:*

url url
version version
client an object of class WPSClient
logger logger type NULL, "INFO" or "DEBUG"
... any other parameter to pass to OWSGetCapabilities service request
**Method** getProcesses(): Get the list of processes offered by the service capabilities. *pretty* allows to control the type output. If *TRUE*, a `data.frame` will be returned. When prettified output, it is also possible to get a ‘full’ description of the process by setting *full = TRUE* in which case a the WPS client will request a process description (with more information about the process) for each process listed in the capabilities.

*Usage:*

```r
WPSCapabilities$getProcesses(pretty = FALSE, full = FALSE)
```

*Arguments:*

- `pretty`: pretty
- `full`: full

*Returns:* a list of `WPSProcessDescription` or a `data.frame`

**Method** describeProcess(): Get the description of a process, given its identifier, returning an object of class `WPSProcessDescription`

*Usage:*

```r
WPSCapabilities$describeProcess(identifier)
```

*Arguments:*

- `identifier`: process identifier

*Returns:* an object of class `WPSProcessDescription`

**Method** execute(): Execute a process, given its identifier

*Usage:*

```r
WPSCapabilities$execute(
  identifier,
  dataInputs = list(),
  responseForm = NULL,
  storeExecuteResponse = FALSE,
  lineage = NULL,
  status = NULL,
  update = FALSE,
  updateInterval = 1
)
```

*Arguments:*

- `identifier`: process identifier
- `dataInputs`: a named list of data inputs, objects of class `WPSLiteralData`, `WPSComplexData` or `WPSBoundingBoxData`
- `responseForm`: response form, object of class `WPSResponseDocument`
- `storeExecuteResponse`: store execute response? object of class logical. FALSE by default
- `lineage`: lineage, object of class logical
- `status`: status, object of class logical
- `update`: update, object of class logical. For asynchronous requests
- `updateInterval`: update interval, object of class integer. For asynchronous requests

**Method** clone(): The objects of this class are cloneable with this method.
Usage:
WPSCapabilities$clone(deep = FALSE)

Arguments:
  deep  Whether to make a deep clone.

Note
Class used to read a WPSCapabilities document. The use of WPSClient is recommended instead to benefit from the full set of capabilities associated to a WPS server.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

WPSClient

Description
WPSClient

Format
R6Class object.

Value
Object of R6Class with methods for interfacing an OGC Web Processing Service.

Super classes
ows4R::OGCAbstractObject -> ows4R::OWSClient -> WPSClient

Methods
Public methods:
- WPSClient$new()
- WPSClient$getCapabilities()
- WPSClient$reloadCapabilities()
- WPSClient$getProcesses()
- WPSClient$describeProcess()
- WPSClient$execute()
- WPSClient$clone()
**Method** `new()`: This method is used to instantiate a **WPSClient** with the `url` of the OGC service. Authentication is supported using basic auth (using `user/pwd` arguments), bearer token (using `token` argument), or custom (using `headers` argument). By default, the `logger` argument will be set to `NULL` (no logger). This argument accepts two possible values: `INFO` to print only **ows4R** logs, `DEBUG` to print more verbose logs

*Usage:*

```r
WPSClient$new(
  url,
  serviceVersion = NULL,
  user = NULL,
  pwd = NULL,
  token = NULL,
  headers = c(),
  config = httr::config(),
  cas_url = NULL,
  logger = NULL
)
```

*Arguments:*

- `url` `url`
- `serviceVersion` WFS service version
- `user` `user`
- `pwd` password
- `token` `token`
- `headers` `headers`
- `config` `config`
- `cas_url` Central Authentication Service (CAS) URL
- `logger` `logger`

**Method** `getCapabilities()`: Get WPS capabilities

*Usage:*

```r
WPSClient$getCapabilities()
```

*Returns:* an object of class **WPSCapabilities**

**Method** `reloadCapabilities()`: Reloads WPS capabilities

*Usage:*

```r
WPSClient$reloadCapabilities()
```

**Method** `getProcesses()`: Get the list of processes offered by the service capabilities. `pretty` allows to control the type output. If `TRUE`, a data.frame will be returned. When prettified output, it is also possible to get a 'full' description of the process by setting `full = TRUE` in which case the WPS client will request a process description (with more information about the process) for each process listed in the capabilities.

*Usage:*

```r
WPSClient$getProcesses(pretty = FALSE, full = FALSE)
```

*Arguments:*

- `pretty` `pretty`
- `full` `full`
pretty pretty full full

Returns: a list of WPSProcessDescription or a data.frame

Method describeProcess(): Get the description of a process, given its identifier, returning an object of class WPSProcessDescription

Usage: WPSClient$describeProcess(identifier)

Arguments:
identifier process identifier

Returns: an object of class WPSProcessDescription

Method execute(): Execute a process, given its identifier

Usage: WPSClient$execute(
  identifier,
  dataInputs = list(),
  responseForm = NULL,
  storeExecuteResponse = FALSE,
  lineage = NULL,
  status = NULL,
  update = FALSE,
  updateInterval = 1
)

Arguments:
identifier process identifier
dataInputs a named list of data inputs, objects of class WPSLiteralData, WPSComplexData or WPSBoundingBoxData
responseForm response form, object of class WPSResponseDocument
storeExecuteResponse store execute response? object of class logical. FALSE by default
lineage lineage, object of class logical
status status, object of class logical
update update, object of class logical. For asynchronous requests
updateInterval update interval, object of class integer. For asynchronous requests

Method clone(): The objects of this class are cloneable with this method.

Usage: WPSClient$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
## WPSComplexData

### Description

WPSComplexData

### Format

*R6Class* object.

### Value

Object of *R6Class* for modelling a WPS Complex Data

### Super class

`ows4R::OGCAbstractObject` $\rightarrow$ WPSComplexData

### Public fields

`value` value

### Methods

Public methods:

- `WPSComplexData$new()`
- `WPSComplexData$decode()`
- `WPSComplexData$checkValidity()`
- `WPSComplexData$getFeatures()`
- `WPSComplexData$clone()`

Method `new()`: initialises an object of class *WPSComplexData*

*Usage:*
WPSComplexData

WPSComplexData$new(
  xml = NULL,
  value = NULL,
  schema = NULL,
  mimeType = NULL,
  serviceVersion = "1.0.0",
  cdata = TRUE
)

Arguments:
xml an object of class XMLInternalNode-class to initialize from XML
value
schema
mimeType mime type
serviceVersion WPS service version
cdata whether value has to be wrapped in a XML CDATA. Default is TRUE

Method decode(): Decodes an object of class WPSComplexData from XML

Usage:
WPSComplexData$decode(xml)

Arguments:
xml an object of class XMLInternalNode-class to initialize from XML

Method checkValidity(): Check the object against a parameter description inherited from a WPS process description, object of class WPSComplexInputDescription. If not valid, the function will raise an error.

Usage:
WPSComplexData$checkValidity(parameterDescription)

Arguments:
parameterDescription object of class WPSComplexInputDescription

Returns: an error if not valid

Method getFeatures(): Get features

Usage:
WPSComplexData$getFeatures()

Returns: object of class sf

Method clone(): The objects of this class are cloneable with this method.

Usage:
WPSComplexData$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
**Description**

WPSComplexInputDescription

**Format**

*R6Class* object.

**Value**

Object of *R6Class* modelling a WPS process complex input description

**Super classes**

ows4R::OGCAbstractObject -> ows4R::WPSParameter -> ows4R::WPSDescriptionParameter

-> ows4R::WPSInputDescription -> WPSComplexInputDescription

**Methods**

**Public methods:**

- WPSComplexInputDescription$new()
- WPSComplexInputDescription$clone()

**Method** `new()`: Initializes a `WPSComplexInputDescription`

*Usage:*

WPSComplexInputDescription$new(xml = NULL, version, logger = NULL, ...)

*Arguments:*

xml object of class XMLInternalNode-class from XML
version WPS service version
logger logger
... any other parameter

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

WPSComplexInputDescription$clone(deep = FALSE)

*Arguments:*

deep Whether to make a deep clone.
**Note**
Class used internally by ows4R

**Author(s)**
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**WPSClассOutputDescription**

**Description**
WPSClассOutputDescription
WPSClассOutputDescription

**Format**
*R6Class* object.

**Value**
Object of *R6Class* modelling a WPS process complex input description

**Super classes**
ows4R::OGCAbstractObject -> ows4R::WPSParameter -> ows4R::WPSDescriptionParameter -> ows4R::WPSOutputDescription -> WPSClассOutputDescription

**Methods**

**Public methods:**

- WPSClассOutputDescription$new()
- WPSClассOutputDescription$clone()

**Method** new(): Initializes a WPSClассOutputDescription

*Usage:*
WPSClассOutputDescription$new(xml = NULL, version, logger = NULL, ...)

*Arguments:*
xml object of class XMLInternalNode-class from XML
version WPS service version
logger logger
... any other parameter

**Method** clone(): The objects of this class are cloneable with this method.
Usage:
WPSComplexOutputDescription$clone(deep = FALSE)

Arguments:
depth Whether to make a deep clone.

Note
Class used internally by ow4R

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Description
WPSDescribeProcess
WPSDescribeProcess

Format
R6Class object.

Value
Object of R6Class for modelling a WPS DescribeProcess request

Super classes
ows4R::OGCAbstractObject -> ow4R::OWSHttpRequest -> WPSDescribeProcess

Methods
Public methods:
• WPSDescribeProcess$new()
• WPSDescribeProcess$clone()

Method new(): Initializes a WPSDescribeProcess service request
Usage:
WPSDescribeProcess$new(
    capabilities,
    op,
    url,
    serviceVersion,
    identifier,
    user = NULL,
    pwd = NULL,
    token = NULL,
    headers = c(),
    config = httr::config(),
    logger = NULL,
    ...
)

Arguments:
- capabilities object of class WPSCapabilities
- op object of class OWSOperation
- url url
- serviceVersion WPS service version
- identifier process identifier
- user user
- pwd password
- token token
- headers headers
- config config
- logger logger
- ... any other parameter to pass to the request

Method clone(): The objects of this class are cloneable with this method.

Usage:
WPSDescribeProcess$clone(deep = FALSE)

Arguments:
- deep Whether to make a deep clone.

Note
Class used internally by ows4R to trigger a WPS DescribeProcess request

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Description

WPSDescriptionParameter

Format

R6Class object.

Value

Object of R6Class modelling a WPS process input description parameter

Super classes

ows4R::OGCAbstractObject -> ows4R::WPSParameter -> WPSDescriptionParameter

Methods

Public methods:

• WPSDescriptionParameter$new()
• WPSDescriptionParameter$getDataType()
• WPSDescriptionParameter$getFormats()
• WPSDescriptionParameter$clone()

Method new(): Initializes a WPSDescriptionParameter

Usage:
WPSDescriptionParameter$new(xml = NULL, version, logger = NULL, ...)

Arguments:
xml object of class XMLInternalNode-class from XML
version WPS service version
logger logger
... any other parameter

Method getDataType(): Get data type

Usage:
WPSDescriptionParameter$getDataType()

Returns: object of class character

Method getFormats(): Get formats
Usage:
WPSDescriptionParameter$getFormats()

Returns: the formats

Method clone(): The objects of this class are cloneable with this method.

Usage:
WPSDescriptionParameter$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Note
Class used internally by ow4R

Author(s)
Emmanuel Blondel <emmanuel.blondell1@gmail.com>

WPSException

Description
WPSException
WPSException

Format
R6Class object.

Value
Object of R6Class for modelling a WPS Exception

Super class
ows4R::OGCAbstractObject -> WPSException

Public fields
value value
percentCompleted percent of completion
Methods

Public methods:

- WPSException$new()
- WPSException$decode()
- WPSException$clone()

Method new(): Initializes a WPSException

Usage:
WPSException$new(xml = NULL, serviceVersion = "1.0.0")

Arguments:
xml object of class XMLInternalNode-class from XML
serviceVersion service version. Default "1.0.0"

Method decode(): Decodes an object of class WPSException from XML

Usage:
WPSException$decode(xml)

Arguments:
xml object of class XMLInternalNode-class from XML

Method clone(): The objects of this class are cloneable with this method.

Usage:
WPSException$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

WPSExecute

Description

WPSExecute

Format

R6Class object.

Value

Object of R6Class for modelling a WPS Execute request
Super classes

ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> WPSExecute

Public fields

Identifier  process identifier
DataInputs  list of WPSInput
ResponseForm  response form

Methods

Public methods:

• WPSExecute$new()
• WPSExecute$getProcessDescription()
• WPSExecute$clone()

Method  new(): Initializes a WPSExecute service request

Usage:
WPSExecute$new(
   capabilities,
   op,
   url,
   serviceVersion,
   identifier,
   dataInputs = list(),
   responseForm = NULL,
   storeExecuteResponse = FALSE,
   lineage = NULL,
   status = NULL,
   user = NULL,
   pwd = NULL,
   token = NULL,
   headers = c(),
   config = httr::config(),
   logger = NULL,
   ...
)

Arguments:
capabilities  object of class WPSCapabilities
op  object of class OWSOperation
url  url
serviceVersion  WPS service version
identifier  process identifier
dataInputs  a named list of data inputs, objects of class WPSLiteralData, WPSComplexData or WPSBoundingBoxData
responseForm response form, object of class **WPSResponseDocument**
storeExecuteResponse store execute response? object of class logical. FALSE by default
lineage lineage, object of class logical
status status, object of class logical
user user
pwd password
token token
headers headers
config config
logger logger
... any other parameter to pass to the request

**Method** getProcessDescription(): Get process description

*Usage:*

```r
WPSExecute$getProcessDescription()
```

*Returns:* an object of class **WPSProcessDescription**

**Method** clone(): The objects of this class are cloneable with this method.

*Usage:*

```r
WPSExecute$clone(deep = FALSE)
```

*Arguments:*

deep Whether to make a deep clone.

**Note**

Class used internally by **ows4R** to trigger a WPS Execute request

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**Description**

WPSExecuteResponse

**Format**

**R6Class** object.
Value
Object of \texttt{R6Class} for modelling a WPS Execute response

Super class
\texttt{ows4R::OGCAbstractObject} -> \texttt{WPSExecuteResponse}

Public fields
\begin{itemize}
  \item process
  \item status
  \item statusLocation
  \item statusHistory
  \item processOutputs
  \item exception
\end{itemize}

Methods

\textbf{Public methods:}

\begin{itemize}
  \item \texttt{WPSExecuteResponse$new()}
  \item \texttt{WPSExecuteResponse$getProcess()}
  \item \texttt{WPSExecuteResponse$getStatus()}
  \item \texttt{WPSExecuteResponse$getStatusLocation()}
  \item \texttt{WPSExecuteResponse$getStatusHistory()}
  \item \texttt{WPSExecuteResponse$getProcessOutputs()}
  \item \texttt{WPSExecuteResponse$getException()}
  \item \texttt{WPSExecuteResponse$decode()}
  \item \texttt{WPSExecuteResponse$update()}
  \item \texttt{WPSExecuteResponse$clone()}
\end{itemize}

\textbf{Method} \texttt{new()}: Initializes a \texttt{WPSExecuteResponse}

\textit{Usage:}
\begin{verbatim}
WPSExecuteResponse$new(xml,
  capabilities,
  processDescription = NULL,
  logger = NULL)
\end{verbatim}

\textit{Arguments:}
\begin{itemize}
  \item xml object of class \texttt{XMLInternalNode-class} from \texttt{XML}
  \item capabilities object of class \texttt{WPSCapabilities}
  \item processDescription process description
  \item logger logger
\end{itemize}
Method `getProcess()`: Get process

Usage:
WPSExecuteResponse$getProcess()

Returns: an object of class `WPSProcess`

Method `getStatus()`: Get status

Usage:
WPSExecuteResponse$getStatus()

Returns: an object of class `WPSStatus`

Method `getStatusLocation()`: Get status location

Usage:
WPSExecuteResponse$getStatusLocation()

Returns: an object of class `character`

Method `getStatusHistory()`: Get status history

Usage:
WPSExecuteResponse$getStatusHistory()

Returns: an object of class `character`

Method `getProcessOutputs()`: Get list of process outputs

Usage:
WPSExecuteResponse$getProcessOutputs()

Returns: a list of outputs

Method `getException()`: Get exception

Usage:
WPSExecuteResponse$getException()

Returns: an object of class `WPSException`

Method `decode()`: Decodes an object of class `WPSExecuteResponse` from XML

Usage:
WPSExecuteResponse$decode(xml, capabilities, processDescription, logger)

Arguments:
xml object of class `XMLInternalNode-class` from `XML`
capabilities object of class `WPSCapabilities`
processDescription process description
logger logger

Method `update()`: Updates status history

Usage:
WPSExecuteResponse$update(verbose = FALSE)
Arguments:
verbose  verbose. Default is FALSE

Method clone(): The objects of this class are cloneable with this method.

Usage:
WPSExecuteResponse$clone(deep = FALSE)

Arguments:
deep  Whether to make a deep clone.

Author(s)
Emmanuel Blondel <emmanuel.blondell1@gmail.com>

---

WPSFormat  WPSFormat

---

Description
WPSFormat
WPSFormat

Format
R6Class object.

Value
Object of R6Class modelling a WPS process input description

Super class
ows4R::OGCAbstractObject -> WPSFormat

Methods

Public methods:
• WPSFormat$new()
• WPSFormat$getMimeType()
• WPSFormat$getEncoding()
• WPSFormat$getSchema()
• WPSFormat$setIsDefault()
• WPSFormat$isDefault()
• WPSFormat$clone()

Method new(): Initializes an object of class WPSFormat
Usage:
WPSFormat$new(xml = NULL, version, logger = NULL, ...)

Arguments:
xml an object of class XMLInternalNode-class to initialize from XML
version WPS service version
logger logger
... any additional parameter

Method getMimeType(): Get mime type
Usage:
WPSFormat$getMimeType()
Returns: object of class character

Method getEncoding(): Get encoding
Usage:
WPSFormat$getEncoding()
Returns: object of class character

Method getSchema(): Get schema
Usage:
WPSFormat$getSchema()
Returns: object of class character

Method setIsDefault(): set is default
Usage:
WPSFormat$setIsDefault(default)
Arguments:
default object of class logical

Method isDefault(): is default
Usage:
WPSFormat$isDefault()
Returns: TRUE if default, FALSE otherwise

Method clone(): The objects of this class are cloneable with this method.
Usage:
WPSFormat$clone(deep = FALSE)
Arguments:
depth Whether to make a deep clone.

Note
Class used internally by ows4R

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
**WPSInput**

---

**Description**

WPSInput

**Format**

`R6Class` object.

**Value**

Object of `R6Class` for modelling a WPS Input

**Super class**

`ows4R::OGCAbstractObject` -> WPSInput

**Public fields**

- Identifier `identifier`
- Data `data`

**Methods**

**Public methods:**

- `WPSInput$new()`
- `WPSInput$clone()`

**Method new():** Initializes a `WPSInput`

*Usage:*

```r
WPSInput$new(xml = NULL, identifier, data, serviceVersion = "1.0.0")
```

*Arguments:*

- `xml` object of class `XMLInternalNode-class` from `XML`
- `identifier`
- `data`
- `serviceVersion` service version. Default "1.0.0"

**Method clone():** The objects of this class are cloneable with this method.

*Usage:*

```r
WPSInput$clone(deep = FALSE)
```

*Arguments:*

- `deep` Whether to make a deep clone.
Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Description
WPSInputDescription

Format
R6Class object.

Value
Object of R6Class modelling a WPS process input description

Super classes
ows4R::OGCAbstractObject -> ows4R::WPSParameter -> ows4R::WPSDescriptionParameter
-> WPSInputDescription

Methods
Public methods:
- WPSInputDescription$new()
- WPSInputDescription$getMinOccurs()
- WPSInputDescription$getMaxOccurs()
- WPSInputDescription$asDataFrame()
- WPSInputDescription$clone()

Method new(): Initializes a WPSInputDescription
Usage:
WPSInputDescription$new(xml = NULL, version, logger = NULL, ...)
Arguments:
xml object of class XMLInternalNode-class from XML
version WPS service version
logger logger
... any other parameter

Method getMinOccurs(): Get min occurs
Usage:
WPSLiteralData

WPSInputDescription$getMinOccurs()
Returns: the min occurs

Method getMaxOccurs(): Get max occurs
Usage:
WPSInputDescription$getMaxOccurs()
Returns: the max occurs

Method asDataFrame(): Get input description as data.frame
Usage:
WPSInputDescription$asDataFrame()
Returns: object of class character

Method clone(): The objects of this class are cloneable with this method.
Usage:
WPSInputDescription$clone(deep = FALSE)
Arguments:
deep Whether to make a deep clone.

Note
Class used internally by ows4R

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

WPSLiteralData  WPSLiteralData

Description
WPSLiteralData
WPSLiteralData

Format
R6Class object.

Value
Object of R6Class for modelling a WPS Literal Data

Super class
ows4R::OGCAbstractObject -> WPSLiteralData
Public fields

value value

Methods

Public methods:

- WPSLiteralData$new()
- WPSLiteralData$decode()
- WPSLiteralData$checkValidity()
- WPSLiteralData$clone()

Method new(): Initializes an object of class WPSLiteralData

Usage:
WPSLiteralData$new(xml = NULL, value = NULL, serviceVersion = "1.0.0")

Arguments:
xml an object of class XMLInternalNode-class to initialize from XML
value value
serviceVersion WPS service version

Method decode(): Decodes an object of class WPSLiteralData from XML

Usage:
WPSLiteralData$decode(xml)

Arguments:
xml an object of class XMLInternalNode-class to initialize from XML

Method checkValidity(): Check the object against a parameter description inherited from a WPS process description, object of class WPSLiteralInputDescription. If not valid, the function will raise an error.

Usage:
WPSLiteralData$checkValidity(parameterDescription)

Arguments:
parameterDescription object of class WPSLiteralInputDescription
Returns: an error if not valid

Method clone(): The objects of this class are cloneable with this method.

Usage:
WPSLiteralData$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Description

WPSLiteralInputDescription

Value

Object of R6Class modelling a WPS process input description

Super classes

ows4R::OGCAbstractObject -> ows4R::WPSParameter -> ows4R::WPSDescriptionParameter -> ows4R::WPSInputDescription -> WPSLiteralInputDescription

Methods

Public methods:

• WPSLiteralInputDescription$new()
• WPSLiteralInputDescription$getDataType()
• WPSLiteralInputDescription$getDefaultValue()
• WPSLiteralInputDescription$getAllowedValues()
• WPSLiteralInputDescription$clone()

Method new(): Initializes a WPSLiteralInputDescription

Usage:
WPSLiteralInputDescription$new(xml = NULL, version, logger = NULL, ...)

Arguments:
xml object of class XMLInternalNode-class from XML
version WPS service version
logger logger
... any other parameter

Method getDataType(): Get data type

Usage:
WPSLiteralInputDescription$getDataType()

Returns: the data type, object of class character
Method `getDefaultValue()`: Get default value

Usage:
WPSLiteralInputDescription$getDefaultValue()

Returns: the default value, object of class character

Method `getAllowedValues()`: Get allowed values

Usage:
WPSLiteralInputDescription$getAllowedValues()

Returns: the allowed values

Method `clone()`: The objects of this class are cloneable with this method.

Usage:
WPSLiteralInputDescription$clone(deep = FALSE)

Arguments:
depth Whether to make a deep clone.

Note
Class used internally by `ows4R`

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

WPSLiteralOutputDescription

Description
WPSLiteralOutputDescription
WPSLiteralOutputDescription

Format
`R6Class` object.

Value
Object of `R6Class` modelling a WPS process literal output description

Super classes
ows4R::OGCAbstractObject -> ows4R::WPSParameter -> ows4R::WPSDescriptionParameter -> ows4R::WPSOutputDescription -> WPSLiteralOutputDescription
Methods

Public methods:

• \texttt{WPSLiteralOutputDescription$new()}
• \texttt{WPSLiteralOutputDescription$clone()}

Method \texttt{new()}: Initializes a \texttt{WPSLiteralOutputDescription}

Usage:
\texttt{WPSLiteralOutputDescription$new(xml = NULL, version, logger = NULL, ...)}

Arguments:
\begin{itemize}
  \item \texttt{xml} object of class \texttt{XMLInternalNode-class} from \texttt{XML}
  \item \texttt{version} WPS service version
  \item \texttt{logger} logger
  \item \ldots\text{ any other parameter}
\end{itemize}

Method \texttt{clone()}: The objects of this class are cloneable with this method.

Usage:
\texttt{WPSLiteralOutputDescription$clone(deep = FALSE)}

Arguments:
\begin{itemize}
  \item \texttt{deep} Whether to make a deep clone.
\end{itemize}

Note

Class used internally by \texttt{ows4R}

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

\begin{center}
<table>
<thead>
<tr>
<th>WPSOutput</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{WPSOutput}</td>
</tr>
</tbody>
</table>
\end{center}

Description

WPSOutput
WPSOutput

Format

\texttt{R6Class} object.

Value

Object of \texttt{R6Class} for modelling a WPS Input
Super class

\textit{ows4R::OGCAbstractObject} \rightarrow \texttt{WPSOutput}

Public fields

- Identifier \texttt{identifier}
- Title \texttt{title}
- Data \texttt{data}

Methods

Public methods:

- \texttt{WPSOutput$new()}
- \texttt{WPSOutput$decode()}
- \texttt{WPSOutput$getData()}
- \texttt{WPSOutput$getDataValue()}
- \texttt{WPSOutput$clone()}

**Method new():** Initializes a \texttt{WPSOutput}

**Usage:**

\begin{verbatim}
WPSOutput$new(
  xml = NULL,
  identifier = NULL,
  title = NULL,
  data = NULL,
  dataType = NULL,
  serviceVersion = "1.0.0"
)
\end{verbatim}

**Arguments:**

- \texttt{xml} object of class \texttt{XMLInternalNode-class} from \texttt{XML}
- \texttt{identifier} identifier
- \texttt{title} title
- \texttt{data} data
- \texttt{dataType} data type
- \texttt{serviceVersion} WPS service version

**Method decode():** Decodes an object of class \texttt{WPSOutput} from \texttt{XML}

**Usage:**

\begin{verbatim}
WPSOutput$decode(xml)
\end{verbatim}

**Arguments:**

- \texttt{xml} object of class \texttt{XMLInternalNode-class} from \texttt{XML}

**Method getData():** Get data

**Usage:**
WPSOutputDescription

WPSOutput$getData()

*Returns:* data

**Method** getDataValue(): Get data value

*Usage:*
WPSOutput$getDataValue()

*Returns:* data value

**Method** clone(): The objects of this class are cloneable with this method.

*Usage:*
WPSOutput$clone(deep = FALSE)

*Arguments:*
depth Whether to make a deep clone.

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**Description**

WPSOutputDescription

WPSOutputDescription

**Format**

*R6Class* object.

**Value**

Object of *R6Class* modelling a WPS process input description

**Super classes**

ows4R::OGCAbstractObject -> ows4R::WSPParameter -> ows4R::WPSDescriptionParameter

-> WPSOutputDescription
Methods

Public methods:

• `WPSOutputDescription$new()`
• `WPSOutputDescription$clone()`

Method `new()`: Initializes a `WPSOutputDescription`

Usage:
`WPSOutputDescription$new(xml = NULL, version, logger = NULL, ...)`

Arguments:
- `xml` object of class `XMLInternalNode-class` from `XML`
- `version` WPS service version
- `logger` logger
- `...` any other parameter

Method `clone()`: The objects of this class are cloneable with this method.

Usage:
`WPSOutputDescription$clone(deep = FALSE)`

Arguments:
- `deep` Whether to make a deep clone.

Note
Class used internally by `ows4R`

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

WPSParameter  WPSParameter

**Description**

WPSParameter

**Format**

`R6Class` object.

**Value**

Object of `R6Class` modelling a WPS parameter
Super class

ows4R::OGCAbstractObject -> WPSPParameter

Methods

Public methods:

• WPSPParameter$new()
• WPSPParameter$getIdentifier()
• WPSPParameter$getTitle()
• WPSPParameter$getAbstract()
• WPSPParameter$clone()

Method new(): Initializes an object of class WPSPParameter

Usage:
WPSPParameter$new(xml = NULL, version, logger = NULL, ...)

Arguments:
xml an object of class XMLInternalNode-class to initialize from XML
version WPS service version
logger logger
... any additional parameter

Method getIdentifier(): Get identifier

Usage:
WPSPParameter$getIdentifier()

Returns: object of class character

Method getTitle(): Get title

Usage:
WPSPParameter$getTitle()

Returns: object of class character

Method getAbstract(): Get abstract

Usage:
WPSPParameter$getAbstract()

Returns: object of class character

Method clone(): The objects of this class are cloneable with this method.

Usage:
WPSPParameter$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.
Note
Abstract class used internally by ows4R

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Value
Object of \texttt{R6Class} modelling a WPS process

Super class
\texttt{ows4R::OGCAbstractObject} -> \texttt{WPSProcess}

Methods

\textbf{Public methods:}

- \texttt{WPSProcess$new()}
- \texttt{WPSProcess$getIdentifer()}
- \texttt{WPSProcess$getTitle()}
- \texttt{WPSProcess$getVersion()}
- \texttt{WPSProcess$getDescription()}
- \texttt{WPSProcess$execute()}
- \texttt{WPSProcess$clone()}

\textbf{Method} \texttt{new():} Initializes a \texttt{WPSProcess}

\textit{Usage:}
\texttt{WPSProcess$new(xml, capabilities = NULL, version, logger = NULL, ...)}

\textit{Arguments:}
xml object of class \texttt{XMLInternalNode-class} from XML
capabilities object of class \texttt{WPSCapabilities}
version service version
logger ... any additional parameter

**Method** getIdentifier(): Get identifier

*Usage:*

```
WPSProcess$getIdentifier()
```

*Returns:*

object of class `character`

**Method** getTitle(): Get title

*Usage:*

```
WPSProcess$getTitle()
```

*Returns:*

object of class `character`

**Method** getVersion(): Get version

*Usage:*

```
WPSProcess$.getVersion()
```

*Returns:*

object of class `character`

**Method** getDescription(): Get description

*Usage:*

```
WPSProcess$getDescription()
```

*Returns:*

object of class `WPSProcessDescription`

**Method** execute(): Execute process

*Usage:*

```
WPSProcess$execute(
  dataInputs = list(),
  responseForm = NULL,
  storeExecuteResponse = FALSE,
  lineage = NULL,
  status = NULL,
  update = FALSE,
  updateInterval = 1
)
```

*Arguments:*

dataInputs a named list of data inputs, objects of class `WPSLiteralData`, `WPSComplexData` or `WPSBoundingBoxData`

responseForm response form, object of class `WPSResponseDocument`

storeExecuteResponse store execute response? object of class `logical`. FALSE by default

lineage lineage, object of class `logical`

status status, object of class `logical`

update update, object of class `logical`. For asynchronous requests

updateInterval update interval, object of class `integer`. For asynchronous requests
Method clone(): The objects of this class are cloneable with this method.

Usage:
WPSProcess$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Note
Class used internally by ows4R

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>

WPSProcessDescription WPSProcessDescription

Description
WPSProcessDescription
WPSProcessDescription

Format
R6Class object.

Value
Object of R6Class modelling a WPS process description

Super class
ows4R::OGCAbstractObject -&gt; WPSProcessDescription

Methods

Public methods:
- WPSProcessDescription$new()
- WPSProcessDescription$getIdentifier()
- WPSProcessDescription$getTitle()
- WPSProcessDescription$getAbstract()
- WPSProcessDescription$getVersion()
- WPSProcessDescription$isStatusSupported()
- WPSProcessDescription$isStoreSupported()
- WPSProcessDescription$getDataInputs()
• WPSProcessDescription$getProcessOutputs()
• WPSProcessDescription$asDataFrame()
• WPSProcessDescription$clone()

Method new(): Initializes an object of class WPSProcessDescription

Usage:
WPSProcessDescription$new(xml, version, logger = NULL, ...)

Arguments:
xmlobject of class XMLInternalNode-class from XML
versionversion
loggerlogger
... any other parameter

Method getIdentifier(): Get process identifier

Usage:
WPSProcessDescription$getIdentifier()

Returns: the identifier, object of class character

Method getTitle(): Get process title

Usage:
WPSProcessDescription$getTitle()

Returns: the title, object of class character

Method getAbstract(): Get process abstract

Usage:
WPSProcessDescription$getAbstract()

Returns: the abstract, object of class character

Method getVersion(): Get process version

Usage:
WPSProcessDescription$getVersion()

Arguments:
the version, object of class character

Method isStatusSupported(): Indicates if the status is supported

Usage:
WPSProcessDescription$isStatusSupported()

Returns: TRUE if supported, FALSE otherwise

Method isStoreSupported(): Indicates if the store is supported

Usage:
WPSProcessDescription$isStoreSupported()

Returns: TRUE if supported, FALSE otherwise
**Method** `getDataInputs()`: Get data inputs

*Usage:*

```r
WPSProcessDescription$getDataInputs()
```

*Returns:*

a list of objects extending `WPSInputDescription`

**Method** `getProcessOutputs()`: Get process outputs

*Usage:*

```r
WPSProcessDescription$getProcessOutputs()
```

*Returns:*

a list of objects extending `WPSOutputDescription`

**Method** `asDataFrame()`: Convenience method to export a process description as `data.frame`

*Usage:*

```r
WPSProcessDescription$asDataFrame()
```

*Returns:*

a `data.frame` giving the process description

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

```r
WPSProcessDescription$clone(deep = FALSE)
```

*Arguments:*

depth Whether to make a deep clone.

**Note**

Class used internally by `ows4R`

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**WPSResponseDocument**

**Description**

`WPSResponseDocument`

**Format**

`R6Class` object.

**Value**

Object of `R6Class` for modelling an OGC WPS response document
Super class

ows4R::OGCAbstractObject -> WPSResponseDocument

Public fields

Output  output property

Methods

Public methods:

• WPSResponseDocument$new()
• WPSResponseDocument$clone()

Method new(): Initializes a WPSResponseDocument

Usage:
WPSResponseDocument$new(
  xml = NULL,
  storeExecuteResponse = FALSE,
  lineage = NULL,
  status = NULL,
  output = NULL,
  serviceVersion = "1.0.0"
)

Arguments:
xml  object of class XMLInternalNode-class from XML
storeExecuteResponse  store execute response, object of class logical
lineage  lineage, object of class logical
status  status, object of class logical
output  output
serviceVersion  WPS service version

Method clone(): The objects of this class are cloneable with this method.

Usage:
WPSResponseDocument$clone(deep = FALSE)

Arguments:
depth  Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>
WPSStatus

Description

WPSStatus

WPSStatus

Format

R6Class object.

Value

Object of R6Class for modelling a WPS Status

Super class

ows4R::OGCAbstractObject -> WPSStatus

Public fields

value  status value

percentCompleted  status percentage of completion

Methods

Public methods:

• WPSStatus$new()
• WPSStatus$decode()
• WPSStatus$getValue()
• WPSStatus$getPercentCompleted()
• WPSStatus$getCreationTime()
• WPSStatus$clone()

Method new(): Initializes a WPSStatus object

Usage:

WPSStatus$new(xml = NULL, serviceVersion = "1.0.0")

Arguments:

xml  an object of class XMLInternalNode-class from XML
serviceVersion  WPS service version. Default is "1.0.0"

Method decode(): Decodes WPS status from XML

Usage:

WPSStatus$decode(xml)
Arguments:
xml an object of class XMLInternalNode-class from XML

Method getValue(): Get status value, among accepted WPS status values defined in the WPS standard: ProcessAccepted, ProcessStarted, ProcessPaused, ProcessSucceeded, ProcessFailed.
Usage:
WPSStatus$getValue()
Returns: value, object of class character

Method getPercentCompleted(): Get percentage of completion
Usage:
WPSStatus$getPercentCompleted()
Returns: the percentage of completion, object of class integer

Method getCreationTime(): Get creation time
Usage:
WPSStatus$getCreationTime()
Returns: the creation time, object of class POSIXct

Method clone(): The objects of this class are cloneable with this method.
Usage:
WPSStatus$clone(deep = FALSE)
Arguments:
deep Whether to make a deep clone.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Index

* Abstract
  OGCAbstractObject, 32
* And
  And, 4
* Authentication
  CASClient, 9
* BBOX
  BBOX, 5
* BinaryComparisonOpType
  BinaryComparisonOpType, 6
  PropertyIsEqualTo, 66
  PropertyIsGreaterThan, 67
  PropertyIsGreaterThanOrEqualTo, 68
  PropertyIsLessThan, 69
  PropertyIsLessThanOrEqualTo, 70
  PropertyIsNotEqualTo, 73
* BinaryLogicOpType
  And, 4
  BinaryLogicOpType, 7
  Or, 39
* Bounding
  WPSBoundingBoxData, 126
* Box
  WPSBoundingBoxData, 126
* CAS
  CASClient, 9
* CSW
  CSWCapabilities, 10
  CSWClient, 11
  CSWDescribeRecord, 18
  CSWGetRecordByld, 20
  CSWGetRecords, 21
  CSWHarvest, 23
  CSWRecordProperty, 27
  CSWTransaction, 29
* Capabilities
  CSWCapabilities, 10
  WPSCapabilities, 127
* Central
  CASClient, 9
* CodeType
  OWSCodeType, 48
* Common
  OWSClient, 45
* ComplexData
  WPSCaplexData, 132
* Coverage
  WCSClient, 79
  WCSCoverage, 82
  WCSCoverageDescription, 84
  WCSCoverageDomain, 86
  WCSCoverageSpatialDomain, 88
  WCSCoverageSummary, 90
  WCSCoverageTemporalDomain, 93
  WCSGridCRS, 99
* DescribeCoverage
  WCSDescribeCoverage, 95
* DescribeFeatureType
  WFSDescribeFeatureType, 106
* DescribeProcess
  WPSCnribeProcess, 136
* DescribeRecord
  CSWDescribeRecord, 18
* Exception
  WPSException, 139
* ExecuteResponse
  WPSExecuteResponse, 142
* Execute
  WPSExecute, 140
* Expression
  And, 4
  BBOX, 5
  BinaryComparisonOpType, 6
  BinaryLogicOpType, 7
  Not, 31
  OGCExpression, 36
  Or, 39
  PropertyIsBetween, 64
WCSClient, 79
WCSCoverage, 82
WCSCoverageDescription, 84
WCSCoverageDomain, 86
WCSCoverageSpatialDomain, 88
WCSCoverageSummary, 90
WCSCoverageTemporalDomain, 93
WCSDescribeCoverage, 95
WCSGetCoverage, 97
WCSGridCRS, 99
WFSCapabilities, 102
WFSClient, 104
WFSDescribeFeatureType, 106
WFSFeatureType, 108
WFSFeatureTypeElement, 111
WFSGetFeature, 113
WMSCapabilities, 115
WMSClient, 117
WMSGetFeatureInfo, 120
WMLayer, 122
WPSBoundingBoxData, 126
WFSCapabilities, 127
WPSClient, 129
WPSComplexData, 132
WPSComplexInputDescription, 134
WPSComplexOutputDescription, 135
WPSDescribeProcess, 136
WPSDescriptionParameter, 138
WPSException, 139
WPSExecute, 140
WPSExecuteResponse, 142
WPSFormat, 145
WPSInput, 147
WPSInputDescription, 148
WPSLiteralData, 149
WPSLiteralInputDescription, 151
WPSLiteralOutputDescription, 152
WPSOutput, 153
WPSOutputDescription, 155
WPSParameter, 156
WPSProcess, 158
WPSProcessDescription, 160
WPSProcessDescriptionParameter, 160
WPSPropertyIsBetween, 64
WPSPropertyIsEqualTo, 66
WPSPropertyIsGreaterThanOrEqualTo, 68
WPSPropertyIsGreaterThan, 67
WPSPropertyIsLessThan, 69
WPSPropertyIsLesserThanOrEqualTo, 70
WPSPropertyIsLike, 71
WPSPropertyIsLesserThanOrEqualTo, 73
WPSPropertyIsNull

* Object
  * OGCAbstractObject, 32
* Or
  * Or, 39
* Parameter
  * WPSParameter, 156
* ProcessDescription
  * WPSProcessDescription, 160
* Processing
  * WPSClient, 129
* Process
  * WPSClient, 129
  * WPSComplexInputDescription, 134
  * WPSComplexOutputDescription, 135
  * WPSDescriptionParameter, 138
  * WPSFormat, 145
  * WPSInputDescription, 148
  * WPSLiteralInputDescription, 151
  * WPSLiteralOutputDescription, 152
  * WPSOutputDescription, 155
  * WPSProcess, 158
  * WPSProcessDescription, 160
* PropertyIsBetween
  * PropertyIsBetween, 64
* PropertyIsEqualTo
  * PropertyIsEqualTo, 66
* PropertyIsGreaterThanOrEqualTo
  * PropertyIsGreaterThanOrEqualTo, 68
* PropertyIsGreaterThan
  * PropertyIsGreaterThan, 67
* PropertyIsLessThan
  * PropertyIsLessThan, 69
* PropertyIsLesserThanOrEqualTo
  * PropertyIsLesserThanOrEqualTo, 70
* PropertyIsLike
  * PropertyIsLike, 71
* PropertyIsLesserThanOrEqualTo
  * PropertyIsLesserThanOrEqualTo, 73
* PropertyIsNull
INDEX

* Property
  - PropertyIsNull, 74
* Provider
  - OWSServiceProvider, 61
* Query
  - CSWQuery, 25
* RecordProperty
  - CSWRecordProperty, 27
* Request
  - OWSHttpRequest, 50
  - OWSRequest, 57
* ResponseDocument
  - WPSResponseDocument, 162
* Service
  - CASClient, 9
  - OWSRequest, 57
  - OWSServiceIdentification, 59
  - OWSServiceProvider, 61
* Status
  - WPSStatus, 164
* Transaction
  - CSWTransaction, 29
* UnaryLogicOpType
  - Not, 31
  - UnaryLogicOpType, 76
* Utils
  - OWSUtils, 62
* WCS
  - WCSCapabilities, 77
  - WCSClient, 79
  - WCSCoverage, 82
  - WCSCoverageDescription, 84
  - WCSCoverageDomain, 86
  - WCSCoverageSpatialDomain, 88
  - WCSCoverageSummary, 90
  - WCSCoverageTemporalDomain, 93
  - WCSDescribeCoverage, 95
  - WCSServiceCoverage, 97
  - WCSGridCRS, 99
  - WCSServiceElevationDomain, 100
  - WCSServiceTimeDomain, 101
* WFS
  - WFSCapabilities, 102
  - WFSClient, 104
  - WFSDescribeFeatureType, 106
  - WFSFeatureType, 108
  - WFSFeatureTypeElement, 111
  - WFSGetFeature, 113
* WMS
  - WMSCapabilities, 115
  - WMSClient, 117
  - WMSGetFeatureInfo, 120
  - WMSLayer, 122
* WPS
  - WPSBoundingBoxData, 126
  - WPSCapabilities, 127
  - WPSClient, 129
  - WPSComplexData, 132
  - WPSComplexInputDescription, 134
  - WPSComplexOutputDescription, 135
  - WPSDescribeProcess, 136
  - WPSDescriptionParameter, 138
  - WPSException, 139
  - WPSExecute, 140
  - WPSExecuteResponse, 142
  - WPSFormat, 145
  - WPSInput, 147
  - WPSInputDescription, 148
  - WPSLiteralData, 149
  - WPSLiteralInputDescription, 151
  - WPSLiteralOutputDescription, 152
  - WPSOutput, 153
  - WPSOutputDescription, 155
  - WPSParameter, 156
  - WPSProcess, 158
  - WPSProcessDescription, 160
  - WPSResponseDocument, 162
  - WPSStatus, 164
* boundingbox
  - OWSBoundingBox, 40
  - OWSWGS84BoundingBox, 63
* catalogue
  - CSWSClient, 11
* complex
  - WPSComplexInputDescription, 134
  - WPSComplexOutputDescription, 135
* crs
  - WCSGridCRS, 99
* data
  - WPSBoundingBoxData, 126
* description
  - WPSComplexInputDescription, 134
  - WPSComplexOutputDescription, 135
  - WPSDescriptionParameter, 138
  - WPSFormat, 145
  - WPSInputDescription, 148
  - WPSLiteralInputDescription, 151
PropertyIsGreaterThanOrEqualTo, 74
registerOWSNamespace, 75
setOWSNamespace, 75
stack, 93
UnaryLogicOpType, 76, 76
WCSCapabilities, 77, 77, 80, 91, 96, 98
WCSClient, 78, 79, 79
WCSCoverage, 82, 82
WCSCoverageDescription, 80, 84, 84, 92
WCSCoverageDomain, 86, 86, 88
WCSCoverageFilenameHandler, 87, 93
WCSCoverageSpatialDomain, 87, 88
WCSCoverageSummary, 78, 90, 90, 98
WCSCoverageTemporalDomain, 87, 93, 94
WCSDescribeCoverage, 95, 95
WCGetCoverage, 81, 92, 97, 97
WCGridCRS, 89, 99, 99
WCSGSElevationDomain, 100
WCSGSTimeDomain, 101
WFSCapabilities, 102, 103, 105, 107, 109, 114
WFSClient, 104, 104
WFSDescribeFeatureType, 106, 107
WFSFeatureType, 103, 106, 108, 109
WFSFeatureTypeElement, 105, 110, 111, 112
WFSGetFeature, 105, 111, 113, 114
WMSCapabilities, 115, 115, 118, 121, 123
WMSClient, 117, 117
WMSGetFeatureInfo, 119, 120, 120, 125
WMSLayer, 116, 118, 122, 123
WPSBoundingBoxData, 126, 126, 128, 131, 141, 159
WPSCapabilities, 127, 127, 130, 137, 141, 143, 144, 158
WPSClient, 127, 129, 130
WPSComplexData, 128, 131, 132, 132, 133, 141, 159
WPSComplexInputDescription, 133, 134
WPSComplexOutputDescription, 135, 135
WPSDescribeProcess, 136, 136
WPSDescribeParameter, 138, 138
WPSException, 139, 140, 144
WPSExecute, 140, 141
WPSExecuteResponse, 142, 143, 144
WPSFormat, 145, 145
WPSInput, 141, 147, 147
WPSInputDescription, 148, 148, 162
WPSLiteralData, 128, 131, 141, 149, 150, 159
WPSLiteralInputDescription, 150, 151, 151
WPSLiteralOutputDescription, 152, 153
WPSOutput, 153, 154
WPSOutputDescription, 155, 156, 162
WPSParameter, 156, 157
WPSProcess, 144, 158, 158
WPSProcessDescription, 128, 131, 142, 159, 160, 161
WPSResponseDocument, 128, 131, 142, 159, 162, 163
WPSStatus, 144, 164, 164