Package ‘ows4R’

October 14, 2022

Version 0.3-1
Date 2022-08-19
Title Interface to OGC Web-Services (OWS)
Maintainer Emmanuel Blondel <emmanuel.blondel1@gmail.com>
Depends R (>= 3.3.0), methods, geometa, keyring
Imports R6, httr, openssl, XML (>= 3.96-1.1), jsonlite, sf, terra, parallel
Suggests testthat, knitr, markdown
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.
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 7.2.1
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 2022-08-19 11:00:06 UTC
## R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>And</td>
<td>4</td>
</tr>
<tr>
<td>BBOX</td>
<td>5</td>
</tr>
<tr>
<td>BinaryComparisonOpType</td>
<td>6</td>
</tr>
<tr>
<td>BinaryLogicOpType</td>
<td>7</td>
</tr>
<tr>
<td>CASClient</td>
<td>9</td>
</tr>
<tr>
<td>CSWCapabilities</td>
<td>10</td>
</tr>
<tr>
<td>CSWClient</td>
<td>11</td>
</tr>
<tr>
<td>CSWConstraint</td>
<td>17</td>
</tr>
<tr>
<td>CSWDescribeRecord</td>
<td>18</td>
</tr>
<tr>
<td>CSWGetRecordById</td>
<td>20</td>
</tr>
<tr>
<td>CSWGetRecords</td>
<td>21</td>
</tr>
<tr>
<td>CSWHarvest</td>
<td>23</td>
</tr>
<tr>
<td>CSWQuery</td>
<td>25</td>
</tr>
<tr>
<td>CSWRecordProperty</td>
<td>27</td>
</tr>
<tr>
<td>CSWTransaction</td>
<td>29</td>
</tr>
<tr>
<td>getOWSNamespace</td>
<td>30</td>
</tr>
<tr>
<td>getOWSNamespaces</td>
<td>31</td>
</tr>
<tr>
<td>Not</td>
<td>31</td>
</tr>
<tr>
<td>OGCAbstractObject</td>
<td>32</td>
</tr>
<tr>
<td>OGCExpression</td>
<td>36</td>
</tr>
<tr>
<td>OGCFilter</td>
<td>37</td>
</tr>
<tr>
<td>Or</td>
<td>39</td>
</tr>
<tr>
<td>ows4R</td>
<td>40</td>
</tr>
<tr>
<td>OWSBoundingBox</td>
<td>40</td>
</tr>
<tr>
<td>OWSCapabilities</td>
<td>42</td>
</tr>
<tr>
<td>OWSClient</td>
<td>45</td>
</tr>
<tr>
<td>OWSCodeType</td>
<td>48</td>
</tr>
<tr>
<td>OWSGetCapabilities</td>
<td>49</td>
</tr>
<tr>
<td>OWSHttpRequest</td>
<td>50</td>
</tr>
<tr>
<td>OWSNamespace</td>
<td>53</td>
</tr>
<tr>
<td>OWSOperation</td>
<td>55</td>
</tr>
<tr>
<td>OWSOperationsMetadata</td>
<td>56</td>
</tr>
<tr>
<td>OWSRequest</td>
<td>57</td>
</tr>
<tr>
<td>OWSServiceIdentification</td>
<td>59</td>
</tr>
<tr>
<td>OWSServiceProvider</td>
<td>61</td>
</tr>
<tr>
<td>OWSSUtils</td>
<td>62</td>
</tr>
<tr>
<td>OWSWGS84BoundingBox</td>
<td>63</td>
</tr>
<tr>
<td>PropertyIsBetween</td>
<td>64</td>
</tr>
<tr>
<td>PropertyIsEqualTo</td>
<td>66</td>
</tr>
<tr>
<td>PropertyIsGreaterThan</td>
<td>67</td>
</tr>
<tr>
<td>PropertyIsGreaterThanOrEqualTo</td>
<td>68</td>
</tr>
<tr>
<td>PropertyIsLessThan</td>
<td>69</td>
</tr>
<tr>
<td>PropertyIsLessThanOrEqualTo</td>
<td>70</td>
</tr>
<tr>
<td>PropertyIsLike</td>
<td>71</td>
</tr>
<tr>
<td>PropertyIsNotEqualTo</td>
<td>73</td>
</tr>
<tr>
<td>PropertyIsNull</td>
<td>74</td>
</tr>
<tr>
<td>Topic</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>registerOWSNamespace</td>
<td>75</td>
</tr>
<tr>
<td>setOWSNamespace</td>
<td>75</td>
</tr>
<tr>
<td>UnaryLogicOpType</td>
<td>76</td>
</tr>
<tr>
<td>WCSCapabilities</td>
<td>77</td>
</tr>
<tr>
<td>WCSClient</td>
<td>79</td>
</tr>
<tr>
<td>WCSCoverage</td>
<td>81</td>
</tr>
<tr>
<td>WCSCoverageDescription</td>
<td>83</td>
</tr>
<tr>
<td>WCSCoverageDomain</td>
<td>85</td>
</tr>
<tr>
<td>WCSCoverageFilenameHandler</td>
<td>87</td>
</tr>
<tr>
<td>WCSCoverageSpatialDomain</td>
<td>87</td>
</tr>
<tr>
<td>WCSCoverageSummary</td>
<td>89</td>
</tr>
<tr>
<td>WCSCoverageTemporalDomain</td>
<td>93</td>
</tr>
<tr>
<td>WCSServiceDescription</td>
<td>94</td>
</tr>
<tr>
<td>WCSGetCoverage</td>
<td>96</td>
</tr>
<tr>
<td>WCSSGridCRS</td>
<td>98</td>
</tr>
<tr>
<td>WCGSElevationDomain</td>
<td>99</td>
</tr>
<tr>
<td>WCGSTimeDomain</td>
<td>100</td>
</tr>
<tr>
<td>WFSCapabilities</td>
<td>101</td>
</tr>
<tr>
<td>WFSClient</td>
<td>103</td>
</tr>
<tr>
<td>WFSDescribeFeatureType</td>
<td>105</td>
</tr>
<tr>
<td>WFSFeatureType</td>
<td>107</td>
</tr>
<tr>
<td>WFSFeatureTypeElement</td>
<td>110</td>
</tr>
<tr>
<td>WFSGetFeature</td>
<td>112</td>
</tr>
<tr>
<td>WMSCapabilities</td>
<td>113</td>
</tr>
<tr>
<td>WMSClient</td>
<td>115</td>
</tr>
<tr>
<td>WMSGetFeatureInfo</td>
<td>118</td>
</tr>
<tr>
<td>WMSLayer</td>
<td>120</td>
</tr>
<tr>
<td>WPSBoundingBoxData</td>
<td>124</td>
</tr>
<tr>
<td>WPSCapabilities</td>
<td>125</td>
</tr>
<tr>
<td>WPSClient</td>
<td>127</td>
</tr>
<tr>
<td>WPSCoordinateOpType</td>
<td>130</td>
</tr>
<tr>
<td>WPSCoordinateInputDescription</td>
<td>132</td>
</tr>
<tr>
<td>WPSCoordinateOutputDescription</td>
<td>133</td>
</tr>
<tr>
<td>WPSDescribeProcess</td>
<td>134</td>
</tr>
<tr>
<td>WPSDescriptionParameter</td>
<td>136</td>
</tr>
<tr>
<td>WPSException</td>
<td>137</td>
</tr>
<tr>
<td>WPSExecute</td>
<td>139</td>
</tr>
<tr>
<td>WPSExecuteResponse</td>
<td>141</td>
</tr>
<tr>
<td>WPSFormat</td>
<td>143</td>
</tr>
<tr>
<td>WPSInput</td>
<td>145</td>
</tr>
<tr>
<td>WPSInputDescription</td>
<td>146</td>
</tr>
<tr>
<td>WPSLiteralData</td>
<td>148</td>
</tr>
<tr>
<td>WPSLiteralInputDescription</td>
<td>149</td>
</tr>
<tr>
<td>WPSLiteralOutputDescription</td>
<td>151</td>
</tr>
<tr>
<td>WPSOutput</td>
<td>152</td>
</tr>
<tr>
<td>WPSOutputDescription</td>
<td>154</td>
</tr>
<tr>
<td>WPSParameter</td>
<td>155</td>
</tr>
<tr>
<td>WPSProcess</td>
<td>156</td>
</tr>
</tbody>
</table>
Description

And

And

Format

R6Class object.

Value

Object of R6Class for modelling an And operator

Super classes

ows4R::OGCAbstractObject -> ows4R::OGCExpression -> 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:
dee... 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")
and <- And$new(expr1, expr2)
and_xml <- and$encode() # see how it looks like in XML
```

Description

BBOX

Format

R6Class object.

Value

Object of R6Class for modelling an 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:**

depth  Whether to make a deep clone.

**Author(s)**

Emmanuel Blondel <emmanuel.blondell1@gmail.com>

**Examples**

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

---

**BinaryComparisonOpType**

**Description**

- BinaryComparisonOpType
- BinaryComparisonOpType

**Format**

- `R6Class` object.

**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>
**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:*

eexprVersion 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:
depth 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:
CSWCapabilities$new(url, version, client = NULL, logger = NULL, ...)

Arguments:
url url
version 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:
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>

CSWClient CSWClient

description

CSWClient
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:

```r
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
headers config
headers config
cas_url Central Authentication Service (CAS) URL
logger logger

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

*Usage:*
```r
CSWClient$getCapabilities()
```

*Returns:* an object of class `CSWCapabilities`

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

*Usage:*
```r
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:*
```r
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:*
```r
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()`: Harves 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
#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
```
CSWConstraint

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

Methods

Public methods:

• CSWConstraint$new()
• CSWConstraint$setServiceVersion()
• CSWConstraint$clone()

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

Usage:

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:

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

Description

CSWDescribeRecord

Format

R6Class object.

Value

Object of R6Class for modelling a CSW DescribeRecord request

Super classes

ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> CSWDescribeRecord
## CSWDescribeRecord

### Methods

**Public methods:**

- `CSWDescribeRecord$new()`
- `CSWDescribeRecord$clone()`

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

```r
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.

```r
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:

```r
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:
deep Whether 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>

---

CSWGetRecords

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(),
cfg = 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
cfg cfg
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>

---

**CSWHarvest**

**Description**

CSWHarvest

**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:
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
cfg 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:
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

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 method ensures proper 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:
depth 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")
query <- CSWQuery$new(constraint = cons)
cons <- CSWConstraint$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/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**

**CSWRecordProperty**

**Description**

CSWRecordProperty

CSWRecordProperty

**Format**

*R6Class* object.

**Value**

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

\texttt{ows4R::OGCAbstractObject -> CSWRecordProperty}

Public fields

- \texttt{wrap}  internal property for XML encoding
- \texttt{Name}  name property for request XML encoding
- \texttt{Value}  property for request XML encoding

Methods

\textbf{Public methods:}

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

\textbf{Method \texttt{new()}}: Initializes an object of class \texttt{CSWRecordProperty}

\textit{Usage:}

\texttt{CSWRecordProperty$new(name, value, cswVersion = "2.0.2")}

\textit{Arguments:}

- \texttt{name}  name
- \texttt{value}  value
- \texttt{cswVersion}  CSW service version. Default is "2.0.2"

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

\textit{Usage:}

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

\textit{Arguments:}

- \texttt{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
```
Description

CSWTransaction

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:
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
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:**
```r
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>
**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()
```

---

**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

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

OGCAbstractObject  OGCAbstractObject

Description

OGCAbstractObject
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:*

```java
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:*

```java
OGCAbstractObject$INFO(text)
```

*Arguments:*

- `text` log message text to be displayed

**Method WARN()**: a basic WARN logger function

*Usage:*

```java
OGCAbstractObject$WARN(text)
```

*Arguments:*

- `text` log message text to be displayed
Method **ERROR()**: a basic ERROR logger function

*Usage:*

```r
OGCAbstractObject$ERROR(text)
```

*Arguments:*

text log message text to be displayed

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

*Usage:*

```r
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:*

```r
OGCAbstractObject$getClassName()
```

*Returns: an object of class character*

Method **getClass()**: Get class

*Usage:*

```r
OGCAbstractObject$getClass()
```

*Returns: an object of class R6Class*

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

*Usage:*

```r
OGCAbstractObject$isFieldInheritedFrom(field)
```

*Arguments:*

field field name
OGCAbstractObject

Returns: object of class R6Class

Method getNamespaceDefinition(): Gets the namespace definition

Usage:
OGCAbstractObject$getNamespaceDefinition(recursive = FALSE)

Arguments:
recursive Get all namespace recursively

Returns: the namespace definitions as named list

Method encode(): Encodes as XML. The addNS . 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 an object of class geometa::INSPIREMetadataValidator, with a proper user API key, should be specified as geometa_inspireValidator argument.

Usage:
OGCAbstractObject$encode(
    addNS = TRUE,
    geometa_validate = TRUE,
    geometa_inspire = FALSE,
    geometa_inspireValidator = NULL
)

Arguments:
addNS 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:
OGCAbstractObject$clone(deep = FALSE)

Arguments:
depth Whether to make a deep clone.

Note
abstract class used by ows4R

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

Description

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>

---

**OGCFilter**

---

**Description**

OGCFilter

---

**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:*

```r
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:*

```r
OGCFilter$setFilterVersion(filterVersion)
```

*Arguments:*

- `filterVersion` OGC filter version

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

*Usage:*

```r
OGCFilter$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
```
### Description

Or

Or

### Format

*R6Class* object.

### Value

Object of *R6Class* for modelling an Or operator

### Super classes

ows4R::OCAbstractObject -> 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:*

deep Whether to make a deep clone.

### Author(s)

Emmanuel Blondel <emmanuel.blondell1@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**

---

**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,
    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>

---

Description

OWSGetCapabilities

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:
Method getOWSVersion(): Get OWS version

Usage:
OWSCapabilities$getOWSVersion()

Returns: an object of class character

Method getRequest(): Get request

Usage:
OWSCapabilities$getRequest()

Returns: an object of class OWSServiceIdentification

Method getServiceIdentification(): Get service identification

Usage:
OWSCapabilities$getServiceIdentification()

Returns: an object of class OWSServiceIdentification

Method getServiceProvider(): Get service provider

Usage:
OWSCapabilities$getServiceProvider()

Returns: an object of class OWSServiceProvider

Method getOperationsMetadata(): Get service operations metadata

Usage:
OWSCapabilities$getOperationsMetadata()

Returns: an object of class OWSOperationsMetadata

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

Usage:
OWSCapabilities$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

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:
$\text{OWSClient}\$new(
  \text{url},
  \text{service},
  \text{serviceVersion},
  \text{user} = \text{NULL},
  \text{pwd} = \text{NULL},
  \text{token} = \text{NULL},
  \text{headers} = \text{c()},
  \text{config} = \text{httr::config()},
  \text{cas_url} = \text{NULL},
  \text{logger} = \text{NULL}
)

Arguments:
\text{url} url
\text{service} service name
\text{serviceVersion} CSW service version
\text{user} user
\text{pwd} password
\text{token} token
\text{headers} headers
\text{config} config
\text{cas_url} Central Authentication Service (CAS) URL
\text{logger} logger

Method \text{getUrl}(): Get URL
Usage:
$\text{OWSClient}\$\text{getUrl}()$
Returns: the url of the service, object of class character

Method \text{getVersion}(): Get version
Usage:
$\text{OWSClient}\$\text{getVersion}()$
Returns: the version of the service, object of class character

Method \text{getCapabilities}(): Get capabilities
Usage:
$\text{OWSClient}\$\text{getCapabilities}()$
Returns: the capabilities, object of class \text{OWSCapabilities}

Method \text{getUser}(): Get user
Usage:
$\text{OWSClient}\$\text{getUser}()$
Returns: the user, object of class character
Method `getPwd()`: Get password

Usage:
OWSClient$getPwd()

Returns: the password, object of class character

Method `getToken()`: Get token

Usage:
OWSClient$getToken()

Returns: the token, object of class character

Method `getHeaders()`: Get headers

Usage:
OWSClient$getHeaders()

Returns: the headers, object of class character

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

Usage:
OWSClient$getConfig()

Returns: the `httr` config, if any

Method `getCASUrl()`: Get CAS URL

Usage:
OWSClient$getCASUrl()

Returns: a CAS URL

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

Usage:
OWSClient$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>
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:

depth 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:**
```r
OWSGetCapabilities$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**

**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:
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
token
headers
config
namedParams
attrs
contentType
mimeType
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:*
Method `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 \texttt{new}(): Initializes an \texttt{OWSNamespace}

\textit{Usage:}

\texttt{OWSNamespace$new(id, uri)}

\textit{Arguments:}

- id id
- uri uri

Method \texttt{getDefinition}(): Get namespace definition

\textit{Usage:}

\texttt{OWSNamespace$getDefinition()}

\textit{Returns:} a named list with id and uri

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

\textit{Usage:}

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

\textit{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
- 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 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` 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

```r
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:
depth Whether to make a deep clone.
```

**Note**

Abstract class used internally by **ows4R**

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**Description**

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

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:
depth 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:**

**Usage:**

```r
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:**

```r
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**

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

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

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

Value

Object of \texttt{R6Class} for modelling an \texttt{PropertyIsEqualTo}

Super classes

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

Methods

Public methods:

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

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

\textit{Usage:}

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

\textit{Arguments:}

\begin{itemize}
  \item \texttt{PropertyName} \hspace{1em} property name
  \item \texttt{Literal} \hspace{1em} literal
  \item \texttt{matchCase} \hspace{1em} match case
\end{itemize}

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

\textit{Usage:}

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

\textit{Arguments:}

\begin{itemize}
  \item \texttt{deep} \hspace{1em} Whether to make a deep clone.
\end{itemize}

Author(s)

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

```
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:

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:

PropertyIsGreaterThan$clone(deep = FALSE)

Arguments:

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

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

Description
PropertyIsGreaterThanOrEqualTo
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.
**PropertyIsLessThan**

*Usage:*

\`PropertyIsGreaterThanOrEqualTo\clone(deep = FALSE)\`

*Arguments:*

- `deep` Whether to make a deep clone.

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

**Examples**

```r
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`:
- `Literal`:
- `matchCase`
PropertyIsLessThanOrEqualTo

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:*

deep  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

**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(): Initializes an object extending PropertyIsLessThanOrEqualTo

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

Arguments:
deep  Whether to make a deep clone.

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

Examples

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**

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

**Description**

`registerOWSNamespace` allows to register a new namespace in **ows4R**

**Usage**

```r
registerOWSNamespace(id, uri, force)
```

**Arguments**

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

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

**Examples**

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

---

**setOWSNamespaces**

**Description**

`setOWSNamespaces`

**Usage**

```r
setOWSNamespaces()
```
UnaryLogicOpType

Description

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:
exprVersion  OGC expression version

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

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

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

**Note**
- abstract super class of all the unary logical operation classes

---

**WCSCapabilities**

**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:**
`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.blondell@gmail.com>

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

Description

WCSClient
WCSClient

Format

R6Class object.

Value

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

Super classes

ows4R::OGCAbstractObject -> ows4R::OWSClient -> WCSClient

Methods

Public methods:

• WCSClient$new()
• WCSClient$getCapabilities()
• WCSClient$reloadCapabilities()
• WCSClient$describeCoverage()
• WCSClient$getCoverage()
• WCSClient$clone()

Method new(): This method is used to instantiate a WCSClient 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:

WCSClient$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,
    ...
)
Arguments:
identifier identifier
bbox  bbox. Default is NULL
crs  crs. Default is NULL
time  time. Default is NULL
format  format. Default is "image/tiff"
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
... any other argument to pass to the WCS GetCoverage request
elevation  elevation. Default is NULL

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

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

## End(Not run)

---

WCSCoverage  WCSCoverage

Description

WCSCoverage
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$getReference()`
• `WCSCoverage$getAbstract()`
• `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

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:

```r
WCSCoverageDescription$getSupportedFormats()
```

Method `getDomain()`: get domain. Applies to WCS 1 coverage descriptions

Usage:

```r
WCSCoverageDescription$getDomain()
```

Method `getRange()`: get range. Applies to WCS 1.0 coverage descriptions

Usage:

```r
WCSCoverageDescription$getRange()
```

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

Usage:

```r
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>

---

**Description**

WCSCoverageDomain

**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()

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:
deep Whether to make a deep clone.

Note

Class used internally by ows4R.

Author(s)

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

Description

WCSCoverageFilenameHandler provides a coverage filename handler for coverage download.

Usage

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>identifier</td>
<td>coverage identifier</td>
</tr>
<tr>
<td>time</td>
<td>time</td>
</tr>
<tr>
<td>elevation</td>
<td>elevation</td>
</tr>
<tr>
<td>bbox</td>
<td>bbox</td>
</tr>
<tr>
<td>format</td>
<td>format</td>
</tr>
</tbody>
</table>

Value

the filename to use for coverage download

Author(s)

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

WCSCoverageSpatialDomain

Description

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
owsVersion OWS version
logger logger

Method getEnvelopes(): Get envelopes. Method that applies to WCS 1.0 only

Usage:
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:
WCSCoverageSpatialDomain$getBoundingBox()

Returns: a list of objects of class OWSBoundingBox

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

Usage:
WCSCoverageSpatialDomain$getGrids()

Returns: a list of of objects of class GMLGrid or GMLRectifiedGrid
**Method** `getGridCRS()`: Get Grid CRS. Method that applies to WCS 1.1 only

*Usage:*

\[
\text{WCSCoverageSpatialDomain}\$\text{getGridCRS()}
\]

*Returns:* a list of objects of class `WCSCGridCRS`

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

*Usage:*

\[
\text{WCSCoverageSpatialDomain}\$\text{clone(}\text{deep = FALSE})
\]

*Arguments:*

deep Whether to make a deep clone.

**Note**

Class used internally by ows4R.

**Author(s)**

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

```r
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:

```r
WCSCoverageSummary$getId()
```

Returns: an object of class character

Method getSubtype(): Get sub type

Usage:

```r
WCSCoverageSummary$getSubtype()
```

Returns: an object of class character

Method getSubtypeParent(): Get sub type parent

Usage:

```r
WCSCoverageSummary$getSubtypeParent()
```
Returns: an object of class character

**Method** `getWGS84BoundingBox()`: Get bounding box

*Usage:*

```r
WCSCoverageSummary$getWGS84BoundingBox()
```

*Returns: an object of class* `OWSWG84BoundingBox`

**Method** `getBoundingBox()`: Get WGS84 bounding box

*Usage:*

```r
WCSCoverageSummary$getBoundingBox()
```

*Returns: an object of class* `OWSBoundingBox`

**Method** `getDescription()`: Get description

*Usage:*

```r
WCSCoverageSummary$getDescription()
```

*Returns: an object of class* `WCSCoverageDescription`

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

*Usage:*

```r
WCSCoverageSummary$getDimensions()
```

*Returns: the list of dimensions*

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

*Usage:*

```r
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. Default is NULL
- `crs`  crs. Default is NULL
- `time`  time. Default is NULL
- `elevation`  elevation. Default is NULL
format format. 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)
filename filename. 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:
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 genric filename han-
  dler 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:
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>
WCSCoverageTemporalDomain

Description
WCSCoverageTemporalDomain

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:*
WCS_coverageTemporalDomain$getInstants()

*Returns:* a list of objects of class POSIXct

**Method** getPeriods(): Get time periods

*Usage:*
WCS_coverageTemporalDomain$getPeriods()

*Returns:* a list of objects of class GMLTimePeriod

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

*Usage:*
WCS_coverageTemporalDomain$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:
depth Whether to make a deep clone.

Author(s)

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

### Description

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.
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 XMLInternalNode-class to initialize from XML
serviceVersion service version
owsVersion OWS version
**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*
```
WCSGridCRS$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>
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.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

WCSGSTimeDomain

Description

WCSGSTimeDomain
WCSGSTimeDomain

Format

R6Class object.

Value

Object of R6Class modelling a WCS geoserver time domain object

Super classes

geometa::geometaLogger -> geometa::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:
dee Whether to make a deep clone.

Note
Class used internally by ows4R.

Author(s)
Emmanuel Blondel <emmanuel.blondel1@gmail.com>
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 url
version 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 expr
effect 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.

Author(s)

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

WFSCClient

Format

R6Class object.

Value

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

Super classes

ows4R::OGCAbstractObject -> ows4R::OWSClient -> WFSClient

Methods

Public methods:

• WFSClient$new()
• WFSClient$getCapabilities()
• WFSClient$reloadCapabilities()
• WFSClient$describeFeatureType()
• WFSClient$getFeatures()
• WFSClient$getFeatureTypes()
• WFSClient$clone()

Method new(): This method is used to instantiate a WFSClient 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:
WFSClient$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 `WFSFeatureTypeElement`

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`

Usage:
WFSClient$getFeatureTypes(pretty = FALSE)

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

Returns: a list of \texttt{WFSFeatureType} or a data.frame

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

**Examples**

#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")

#get 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
Value

Object of \texttt{R6Class} for modelling a WFS DescribeFeatureType request

Super classes

\texttt{ows4R::OGCAbstractObject} \rightarrow \texttt{ows4R::OWSHttpRequest} \rightarrow \texttt{WFSDescribeFeatureType}

Methods

Public methods:

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

Method \texttt{new()}: Initializes a \texttt{WFSDescribeFeatureType} service request

Usage:

\begin{verbatim}
WFSDescribeFeatureType$new(
    capabilities,
    op,
    url,
    version,
    typeName,
    user = NULL,
    pwd = NULL,
    token = NULL,
    headers = c(),
    config = httr::config(),
    logger = NULL,
    ...
)
\end{verbatim}

Arguments:

- \texttt{capabilities} an object of class \texttt{WFSCapabilities}
- \texttt{op} object of class \texttt{OWSOperation} as retrieved from capabilities
- \texttt{url} url
- \texttt{version} service version
- \texttt{typeName} typeName
- \texttt{user} user
- \texttt{pwd} pwd
- \texttt{token} token
- \texttt{headers} headers
- \texttt{config} config
- \texttt{logger} logger
- ... any parameter to pass to the service request

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

Usage:
WFSFeatureType

WFSDescribeFeatureType$clone(deep = FALSE)

Arguments:

deept 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**

---

**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$getFeatures()
• WFSFeatureType$clone()

**Method** `new()`: Initializes 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:
ocject of class character

**Method** `getTitle()`: Get feature type title

Usage:
WFSFeatureType$getTitle()

Arguments:
ocject of class character

**Method** `getAbstract()`: Get feature type abstract

Usage:
WFSFeatureType$getAbstract()

Arguments:
ocject of class character

**Method** `getKeywords()`: Get feature type keywords

Usage:
WFSFeatureType$getKeywords()

Arguments:
ocject of class character

**Method** `getDefaultCRS()`: Get feature type default CRS

Usage:
WFSFeatureType$getDefaultCRS()
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 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**

Internal class used by ows4R

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**WFSFeatureTypeElement**

**WFSFeatureTypeElement**

**Description**

WFSFeatureTypeElement

WFSFeatureTypeElement

**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$clone()`

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

*Usage:*

`WFSFeatureTypeElement$new(xmlObj)`

*Arguments:*

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

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

*Usage:*

`WFSFeatureTypeElement$getMinOccurs()`

*Returns:* an object of class `character`
Method getMaxOccurs(): get max occurs

Usage:
WFSFeatureTypeElement$getMaxOccurs()

Returns: an object of class character

Method isNillable(): get if nillable

Usage:
WFSFeatureTypeElement$isNillable()

Returns: an object of class logical

Method getName(): get name

Usage:
WFSFeatureTypeElement$getName()

Returns: an object of class character

Method getType(): get type

Usage:
WFSFeatureTypeElement$getType()

Returns: an object of class character

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:*

```r
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
WMSCapabilities

url
version
typeName
outputFormat
user
password
token
headers
config
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:
deepl 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>

WMSCapabilities

Description
WMSCapabilities
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)"
**WMSClient**

*Arguments:*

expr  expr
exact  exact matching? Default is TRUE

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

*Usage:*

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

*Arguments:*

depth 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>

---

**WMSClient**  **WMSClient**

**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,  
  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:*  
```r
WMSClient$reloadCapabilities()
```

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

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

#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

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

<table>
<thead>
<tr>
<th>WMSGetFeatureInfo</th>
<th>WMSGetFeatureInfo</th>
</tr>
</thead>
</table>

Description
WMSGetFeatureInfo
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,
  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
cconfig config
logger logger
... any parameter to pass to the service request

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

Usage:
WMSGetFeatureInfo$clone(deep = FALSE)

Arguments:
deepl Whether to make a deep clone.

Note
Abstract class used by ows4R to trigger a WMS GetFeatureInfo request

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

| WMSLayer | 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:

```
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:

```
WMSLayer$getName()
```

Returns: object of class character

Method `getTitle()`:

Get layer title

Usage:

```
WMSLayer$getTitle()
```

Returns: object of class character

Method `getAbstract()`:

Get layer abstract

Usage:

```
WMSLayer$getAbstract()
```

Returns: object of class character
Method getKeywords(): Get layer keywords
Usage:
WMSLayer$getKeywords()
Returns: object of class character

Method getDefaultCRS(): Get layer default CRS
Usage:
WMSLayer$getDefaultCRS()
Returns: object of class character

Method getBoundingBox(): Get layer bounding box
Usage:
WMSLayer$getBoundingBox()
Returns: object of class matrix

Method getBoundingBoxSRS(): Get layer bounding box SRS
Usage:
WMSLayer$getBoundingBoxSRS()
Returns: object of class character

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.
WPSBoundingBoxData

Note
Abstract class used by ows4R

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

---

WPSBoundingBoxData WPSBoundingBoxData

Description
WPSBoundingBoxData
WPSBoundingBoxData

Format
R6Class object.

Value
Object of R6Class for modelling an OGC WPS BoundingBox data

Super classes
ows4R::OCSAbstractObject -> 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:
xm1 an object of class XMLInternalNode-class to initialize from XML
owsVersion OWS version
WPSCapabilities

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:
depth Whether to make a deep clone.

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

---

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:
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:
WPSCapabilities$describeProcess(identifier)

Arguments:
identifier  process identifier

Returns: an object of class WPSProcessDescription

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

Usage:
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>
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:**

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 a the WPS client will request a process description (with more information about the process) for each process listed in the capabilities.

**Usage:**

WPSClient$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:**

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:
depth  Whether to make a deep clone.

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

Examples

# example based on a WPS endpoint responding at http://localhost:8080/geoserver/wps
wps <- WPSClient$new("http://localhost:8080/geoserver/wps", serviceVersion = "1.0.0")

caps <- wps$getCapabilities()

Description
WPSComplexData
WPSComplexData

Format
R6Class object.

Value
Object of R6Class for modelling a WPS Complex Data

Super class
ows4R::OGCAbstractObject -> WPSComplexData

Public fields
value value
Methods

Public methods:

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

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

*Usage:*

```r
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` value
- `schema` 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:*

```r
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:*

```r
WPSComplexData$checkValidity(parameterDescription)
```

*Arguments:*

- `parameterDescription` object of class `WPSComplexInputDescription`

*Returns:* an error if not valid

**Method getFeatures():** Get features

*Usage:*
WPSComplexInputDescription

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:
depth Whether to make a deep clone.

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

WPSComplexInputDescription

Description

WPSComplexInputDescription

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

Usage:
WPSComplexInputDescription$clone(deep = FALSE)

Arguments:
deepeWhether to make a deep clone.

Note
Class used internally by ow4R

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

Public methods:

- WPSComplexOutputDescription$new()
- WPSComplexOutputDescription$clone()

Method new(): Initializes a WPSComplexOutputDescription

Usage:
WPSComplexOutputDescription$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:
deep Whether to make a deep clone.

Note

Class used internally by ows4R

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 -> ows4R::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

Abstract class used by ows4R to trigger a WPS DescribeProcess request

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

WPSDescriptionParameter

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 **ows4R**

**Author(s)**

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

---

**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>
**Description**

WPSEexecute

**Format**

R6Class object.

**Value**

Object of R6Class for modelling a WPS Execute request

**Super classes**

ows4R::OGCAbstractObject -> ows4R::OWSHttpRequest -> WPSEexecute

**Public fields**

Identifier  process identifier  
DataInputs  list of WPSInput  
ResponseForm  response form

**Methods**

**Public methods:**

- WPSEexecute$new()
- WPSEexecute$GetProcessDescription()
- WPSEexecute$clone()

**Method new():**  Initializes a WPSEexecute service request

*Usage:*

```
WPSEexecute$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:
WPSExecute$getProcessDescription()
Returns: an object of class WPSProcessDescription

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

Note
Abstract class used by ows4R to trigger a WPS Execute request

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

WPSEexecuteResponse

**Format**

**R6Class** object.

**Value**

Object of **R6Class** for modelling a WPS Execute response

**Super class**

ows4R::OGCAbstractObject -> WPSEexecuteResponse

**Public fields**

process process
status status
statusLocation status location
statusHistory status history
processOutputs process outputs
exception exception

**Methods**

**Public methods:**

- WPSEexecuteResponse$new()
- WPSEexecuteResponse$getProcess()
- WPSEexecuteResponse$getStatus()
- WPSEexecuteResponse$getStatusLocation()
- WPSEexecuteResponse$getStatusHistory()
- WPSEexecuteResponse$getProcessOutputs()
- WPSEexecuteResponse$getException()
- WPSEexecuteResponse$decode()
- WPSEexecuteResponse$update()
- WPSEexecuteResponse$clone()

**Method** new(): Initializes a WPSEexecuteResponse
Usage:
WPSExecuteResponse$new(
  xml,
  capabilities,
  processDescription = NULL,
  logger = NULL
)

Arguments:
xml object of class XMLInternalNode-class from XML
capabilities object of class WPSCapabilities
processDescription process description
logger logger

Method getProcess(): Get process
Usage:
WPSExecuteResponse$getProcess()
Returns: an object of class WPSProcess

Method getStatus(): Get status
Usage:
WPSExecuteResponsegetStatus()
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.blondel1@gmail.com>
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:
WPSInput

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:
deep Whether to make a deep clone.

Note
Class used internally by ow4R

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

Description
WPSInput
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:
WPSInput$new(xml = NULL, identifier, data, serviceVersion = "1.0.0")

Arguments:
xml object of class XMLInternalNode-class from XML
identifier identifier
data data
serviceVersion service version. Default "1.0.0"

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

Usage:
WPSInput$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

WPSInputDescription WPSInputDescription

Description

WPSInputDescription
WPSInputDescription

Format

R6Class object.

Value

Object of R6Class modelling a WPS process input description

Super classes

ows4R::OGCAbstractObject -> ows4R::WSPParameter -> 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:
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>
Description

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>

---

`WPSLiteralInputDescription`

**Description**

`WPSLiteralInputDescription`
`WPSLiteralInputDescription`

**Format**

`R6Class` object.

**Value**

Object of `R6Class` modelling a WPS process input description

**Super classes**

`ows4R::OGCAbstractObject` -> `ows4R::WPSParameter` -> `ows4R::WPSDescriptionParameter` -> `ows4R::WPSInputDescription` -> `WPSLiteralInputDescription`
Methods

Public methods:

- \texttt{WPSLiteralInputDescription$new()}
- \texttt{WPSLiteralInputDescription$getDataType()}
- \texttt{WPSLiteralInputDescription$getDefaultValue()}
- \texttt{WPSLiteralInputDescription$getAllowedValues()}
- \texttt{WPSLiteralInputDescription$clone()}

Method \texttt{new()}:

Usage:
\texttt{WPSLiteralInputDescription$new(xml = NULL, version, logger = NULL, ...)}

Arguments:
- \texttt{xml} object of class \texttt{XMLInternalNode-class} from \texttt{XML}
- \texttt{version} WPS service version
- \texttt{logger} logger
- \ldots any other parameter

Method \texttt{getDataType()}:

Usage:
\texttt{WPSLiteralInputDescription$getDataType()}

Returns: the data type, object of class \texttt{character}

Method \texttt{getDefaultValue()}:

Usage:
\texttt{WPSLiteralInputDescription$getDefaultValue()}

Returns: the default value, object of class \texttt{character}

Method \texttt{getAllowedValues()}:

Usage:
\texttt{WPSLiteralInputDescription$getAllowedValues()}

Returns: the allowed values

Method \texttt{clone()}:

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

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

Note

Class used internally by \texttt{ows4R}

Author(s)

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

Description

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:

- *WPSLiteralOutputDescription*$new()
- *WPSLiteralOutputDescription*$clone()

Method *new()*: Initializes a *WPSLiteralOutputDescription*

Usage:

WPSLiteralOutputDescription$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:

WPSLiteralOutputDescription$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>

WPSOutput

WPSOutput

| WPSOutput | WPSOutput |

Description
WPSOutput
WPSOutput

Format
R6Class object.

Value
Object of R6Class for modelling a WPS Input

Super class
ows4R::OGCAbstractObject -> WPSOutput

Public fields
Identifier identifier
Title tile
Data data

Methods
Public methods:
- WPSOutput$new()
- WPSOutput$decode()
- WPSOutput$getData()
- WPSOutput$getDataValue()
- WPSOutput$clone()

Method new(): Initializes a WPSOutput

Usage:
WPSOutput$new(
  xml = NULL,
  identifier = NULL,
  title = NULL,
  data = NULL,
  dataType = NULL,
  serviceVersion = "1.0.0"
)

Arguments:
xml object of class XMLInternalNode-class from XML
identifier identifier
title title
data data
dataType data type
serviceVersion WPS service version

Method decode(): Decodes an object of class WPSOutput from XML
Usage:
WPSOutput$decode(xml)
Arguments:
xml object of class XMLInternalNode-class from XML

Method getData(): Get data
Usage:
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:
deep 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::WPSParameter -> 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
**Description**

WPSParameter

**Format**

R6Class object.

**Value**

Object of R6Class modelling a WPS parameter

**Super class**

ows4R::OGCAbstractObject -> WPSParameter

**Methods**

**Public methods:**

- WPSParameter$new()
- WPSParameter$getIdentifier()
- WPSParameter$getTitle()
- WPSParameter$getAbstract()
- WPSParameter$clone()

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

**Usage:**

WPSParameter$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:**

WPSParameter$getIdentifier()
Returns: object of class character

Method getTitle(): Get title
Usage:
WPSParameter$getTitle()
Returns: object of class character

Method getAbstract(): Get abstract
Usage:
WPSParameter$getAbstract()
Returns: object of class character

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

Note
Abstract class used internally by ows4R

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

WPSProcess

Description
WPSProcess
WPSProcess

Format
R6Class object.

Value
Object of R6Class modelling a WPS process

Super class
ows4R::OGCAbstractObject -> WPSProcess
Methods

Public methods:
• WPSProcess$new()
• WPSProcess$getIdentifier()
• WPSProcess$getTitle()
• WPSProcess$getVersion()
• WPSProcess$getDescription()
• WPSProcess$execute()
• WPSProcess$clone()

Method new(): Initializes a WPSProcess

Usage:
WPSProcess$new(xml, capabilities = NULL, version, logger = NULL, ...)

Arguments:
xml object of class XMLInternalNode-class from XML
capabilities object of class WPSCapabilities
version service version
logger 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

Description

WPSProcessDescription
WPSProcessDescription

Format

R6Class object.
Value

Object of \texttt{R6Class} modelling a WPS process description

Super class

\texttt{ows4R::OGCAbstractObject} \rightarrow \texttt{WPSProcessDescription}

Methods

Public methods:

- \texttt{WPSProcessDescription$new()}
- \texttt{WPSProcessDescription$getIdentifier()}
- \texttt{WPSProcessDescription$getTitle()}
- \texttt{WPSProcessDescription$getAbstract()}
- \texttt{WPSProcessDescription$getVersion()}
- \texttt{WPSProcessDescription$isStatusSupported()}
- \texttt{WPSProcessDescription$isStoreSupported()}
- \texttt{WPSProcessDescription$getDataInputs()}
- \texttt{WPSProcessDescription$getProcessOutputs()}
- \texttt{WPSProcessDescription$asDataFrame()}
- \texttt{WPSProcessDescription$clone()}

Method \texttt{new()}: Initializes an object of class \texttt{WPSProcessDescription}

Usage:

\texttt{WPSProcessDescription$new(xml, version, logger = NULL, ...)}

Arguments:

- \texttt{xml} object of class \texttt{XMLInternalNode-class} from \texttt{XML}
- \texttt{version} version
- \texttt{logger} logger
- \texttt{...} any other parameter

Method \texttt{getIdentifier()}: Get process identifier

Usage:

\texttt{WPSProcessDescription$getIdentifier()}

Returns: the identifier, object of class \texttt{character}

Method \texttt{getTitle()}: Get process title

Usage:

\texttt{WPSProcessDescription.getTitle()}

Returns: the title, object of class \texttt{character}

Method \texttt{getAbstract()}: Get process abstract

Usage:

\texttt{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:
WPSProcessDescription$getDataInputs()

Returns: a list of objects extending WPSInputDescription

Method getProcessOutputs(): Get process outputs

Usage:
WPSProcessDescription$getProcessOutputs()

Returns: a list of objects extending WPSOutputDescription

Method asDataFrame(): Convenience method to export a process description as data.frame

Usage:
WPSProcessDescription$asDataFrame()

Returns: a data.frame giving the process description

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

Usage:
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>
Description

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

Usage:
WPSResponseDocument$clone(deep = FALSE)

Arguments:
  deep Whether to make a deep clone.

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

| WPSStatus | WPSSstatus |

Description
  WPSStatus
  WPSSstatus

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.blondell1@gmail.com>
Index

* Abstract
  OGCAbstractObject, 32
* And
  And, 4
* Authentication
  CASClient, 9
* BBOX
  BBOX, 5
* BinaryComparisonOpType
  BinaryComparisonOpType, 6
  PropertyIsEqualTo, 66
  PropertyIsGreaterThanOrEqualTo, 68
  PropertyIsLessThanOrEqualTo, 70
  PropertyIsNotEqualTo, 73
* BinaryLogicOpType
  And, 4
  BinaryLogicOpType, 7
  Or, 39
* Bounding
  WPSBoundingBoxData, 124
* Box
  WPSBoundingBoxData, 124
* CAS
  CASClient, 9
* CSW
  CSWCapabilities, 10
  CSWClient, 11
  CSWDescribeRecord, 18
  CSWGetRecordByld, 20
  CSWGetRecords, 21
  CSWHarvest, 23
  CSWRecordProperty, 27
  CSWTransaction, 29
* Capabilities
  CSWCapabilities, 10
  WPSCapabilities, 125
* Central
  CASClient, 9
* CodeType
  OWSCodeType, 48
* Common
  OWSClient, 45
* ComplexData
  WPSCaplexData, 130
* Coverage
  WCSClient, 79
  WCSCoverage, 81
  WCSCoverageDescription, 83
  WCSCoverageDomain, 85
  WCSCoverageSummary, 89
  WCSCoverageTemporalDomain, 93
  WCSGridCRS, 98
* DescribeCoverage
  WCSDescribeCoverage, 94
* DescribeFeatureType
  WFSDescribeFeatureType, 105
* DescribeProcess
  WPSCdescribeProcess, 134
* DescribeRecord
  CSWDescribeRecord, 18
* Exception
  WPSException, 137
* ExecuteResponse
  WPSEXecuteResponse, 141
* Execute
  WPSEXecute, 139
* Expression
  And, 4
  BBOX, 5
  BinaryComparisonOpType, 6
  BinaryLogicOpType, 7
  Not, 31
  OGCExpression, 36
  Or, 39
  PropertyIsBetween, 64
INDEX

PropertyIsEqualTo, 66
PropertyIsGreaterThan, 67
PropertyIsGreaterThanOrEqualTo, 68
PropertyIsLessThan, 69
PropertyIsLessThanOrEqualTo, 70
PropertyIsLike, 71
PropertyIsNotEqualTo, 73
PropertyIsNull, 74
UnaryLogicOpType, 76

* FeatureType
  WFSFeatureType, 107
  WFSFeatureTypeElement, 110

* Feature
  WFSClient, 103

* Filter
  CSWConstraint, 17
  OGCFilter, 37

* GeoServer
  WCSGSElevationDomain, 99
  WCSGSTimeDomain, 100

* GetCapabilities
  OWSCapabilities, 42
  OWSGetCapabilities, 49
  CSWCapabilities, 77
  WFSCapabilities, 101
  WMSCapabilities, 113

* GetCoverage
  WCSGetCoverage, 96

* GetFeatureInfo
  WMSClient, 115
  WMSGetFeatureInfo, 118

* GetFeature
  WFSGetFeature, 112

* GetRecordById
  CSWGetRecordById, 20

* GetRecords
  CSWGetRecords, 21

* HTTP
  OWSHttpRequest, 50

* Harvest
  CSWHarvest, 23

* Identification
  OWSServiceIdentification, 59

* Input
  WPSInput, 145
  WPSOutput, 152

* Layer
  WMSLayer, 120

* LiteralData
  WPSLiteralData, 148

* Map
  WMSClient, 115

* Not
  Not, 31

* OGC
  And, 4
  BBOX, 5
  BinaryComparisonOpType, 6
  BinaryLogicOpType, 7
  CSWCapabilities, 10
  CSWCClient, 11
  CSWConstraint, 17
  CSWDescribeRecord, 18
  CSWGetRecordById, 20
  CSWGetRecords, 21
  CSWHarvest, 23
  CSWQuery, 25
  CSWTransaction, 29
  Not, 31
  OGCAbstractObject, 32
  OGCExpression, 36
  OGCFilter, 37
  Or, 39
  OWSBoundingBox, 40
  OWSCapabilities, 42
  OWSClient, 45
  OWSGetCapabilities, 49
  OWSHttpRequest, 50
  OWSOperation, 55
  OWSOperationsMetadata, 56
  OWSRequest, 57
  OWSServiceIdentification, 59
  OWSServiceProvider, 61
  OWSUtils, 62
  OWSWGS84BoundingBox, 63
  PropertyIsBetween, 64
  PropertyIsEqualTo, 66
  PropertyIsGreater Than, 67
  PropertyIsGreaterThanOrEqualTo, 68
  PropertyIsLessThan, 69
  PropertyIsLessThanOrEqualTo, 70
  PropertyIsLike, 71
  PropertyIsNotEqualTo, 73
  PropertyIsNull, 74
  UnaryLogicOpType, 76
  WSCapabilities, 77
WPSLiteralOutputDescription, 151
WPSOutputDescription, 154
* domain
  WCSCoverageDomain, 85
  WCSCoverageSpatialDomain, 87
  WCSCoverageTemporalDomain, 93
  WCSSGSElevationDomain, 99
  WCSGSTimeDomain, 100
* elevation
  WCSSGSElevationDomain, 99
* grid
  WCSSGridCRS, 98
* input
  WPSComplexInputDescription, 132
  WPSDescriptionParameter, 136
  WPSFormat, 143
  WPSInputDescription, 146
  WPSLiteralInputDescription, 149
* literal
  WPSLiteralOutputDescription, 151
* metadata
  OWSOperationsMetadata, 56
* namespace
  OWSNamespace, 53
* operation
  OWSOperation, 55
  OWSOperationsMetadata, 56
* output
  WPSComplexOutputDescription, 133
  WPSLiteralOutputDescription, 151
  WPSOutputDescription, 154
* parameter
  WPSDescriptionParameter, 136
* service
  CSWClient, 11
* spatial
  WCSCoverageSpatialDomain, 87
* temporal
  WCSCoverageTemporalDomain, 93
* time
  WCSSGSTimeDomain, 100
* web
  CSWClient, 11
* wgs84
  OWSWGS84BoundingBox, 63

And, 4, 4
BBOX, 5, 5

BinaryComparisonOpType, 6, 7
BinaryLogicOpType, 7, 8
CASClient, 9, 9
CSWCapabilities, 10, 11, 13, 19, 21, 22, 24, 29
CSWClient, 11, 11
CSWConstraint, 14, 15, 17, 19
CSWDescribeRecord, 13, 18, 19
CSWGetRecordById, 13, 20, 20, 22
CSWGetRecords, 14, 21
CSWHarvest, 23, 24
CSWQuery, 14, 16, 22, 25
CSWRecordProperty, 14, 15, 27, 28
CSWTransaction, 14, 29, 29

geometa::geometalLogger, 84, 99, 100
gexmeta::GMLAbstractCoverage, 84
gexmeta::GMLAbstractFeature, 84
gexmeta::GMLAbstractObject, 84
gexmeta::GMLCOVAbstractCoverage, 84
gexmeta::ISOAbstractObject, 84, 99, 100
getOWSNamespace, 30
getOWSNamespace, 31
GMLEnvelope, 5, 88
GMLEnvelopeWithTimePeriod, 88
GMLGrid, 88
GMLRectifiedGrid, 88
GMLTimePeriod, 94

ISOResponsibieParty, 62

mclapply, 109

Not, 31, 32

OGCAbstractObject, 32, 34
OGCExpression, 4, 8, 32, 36, 36, 38, 39, 76
OGCFilter, 17, 37, 38
Oc, 39, 39
ows4R, 40
ows4R-package (ows4R), 40
ows4R::BinaryComparisonOpType, 66–70, 73
ows4R::BinaryLogicOpType, 4, 39
ows4R::OGCAbstractObject, 4–6, 8, 10, 12,
  17, 18, 20, 22, 23, 25, 28, 29, 31, 36,
  37, 39, 40, 42, 45, 48–50, 57, 63,
  65–70, 72–74, 76, 77, 79, 82, 85, 88,
  89, 93, 94, 96, 98, 101, 103, 106,
INDEX

107, 112, 113, 115, 118, 120, 124, 125, 127, 130, 132, 133, 135–137, 139, 141, 143, 145, 146, 148, 149, 151, 152, 154–156, 159, 161, 162
ows4R::OGCExpression, 4–6, 8, 31, 39, 65–70, 72–74, 76
ows4R::OWSBoundingBox, 63, 124
ows4R::OWSCapabilities, 10, 77, 101, 113, 125
ows4R::OWSClient, 12, 79, 103, 115, 127
ows4R::OWSHttpRequest, 18, 20, 22, 23, 29, 49, 94, 96, 106, 112, 118, 135, 139
ows4R::UnaryLogicOpType, 31
ows4R::WPSDescriptionParameter, 132, 133, 146, 149, 151, 154
ows4R::WPSInputDescription, 132, 149
ows4R::WPSOutputDescription, 133, 151
ows4R::WPSParameter, 132, 133, 136, 146, 149, 151, 154
OWSBoundingBox, 40, 41, 64, 88, 91
OWSCapabilities, 42, 42, 46, 51, 52, 58
OWSClient, 43, 45
OWSCodeType, 48, 48
OWSGetCapabilities, 11, 43, 44, 49, 49, 78, 102, 114, 126
OWSHttpRequest, 50
OWSNamespace, 53, 54
OWSOperation, 19, 21, 22, 24, 30, 51, 55, 55, 57, 95, 97, 106, 112, 119, 135, 140
OWSOperationsMetadata, 44, 56, 56
OWSRequest, 57, 58, 114
OWSServiceIdentification, 44, 59, 59
OWSServiceProvider, 44, 61, 61
OWSUtils, 62
OWSWGS84BoundingBox, 63, 91
PropertyIsBetween, 64
PropertyIsEqualTo, 66, 66
PropertyIsGreaterThan, 67, 67
PropertyIsGreaterThanOrEqualTo, 68, 68
PropertyIsLessThan, 69, 69
PropertyIsLessThanOrEqualTo, 70, 71
PropertyIsLike, 65, 71, 72, 74
PropertyIsNotEqualTo, 73, 73
PropertyIsNull, 74
R6Class, 4–6, 8–11, 17, 18, 20–23, 25, 27, 29, 31, 32, 36, 37, 39, 40, 42, 45, 48–50, 53, 55–57, 59, 61–64, 66–71, 73, 74,
registerOWSNamespace, 75
setOWSNamespace, 75
SpatRaster, 81
stack, 92
UnaryLogicOpType, 76, 76
WCSCapabilities, 77, 77, 80, 90, 95, 97
WCSClient, 78, 79, 79
WCSCoverage, 81, 82
WCSCoverageDescription, 80, 83, 84, 91
WCSCoverageDomain, 85, 86, 88
WCSCoverageFilenameHandler, 87, 92
WCSCoverageSpatialDomain, 86, 87
WCSCoverageSummary, 78, 89, 90, 97
WCSCoverageTemporalDomain, 86, 93, 93
WCSDescribeCoverage, 94, 95
WCSGetCoverage, 92, 96, 96
WCSGridCRS, 89, 98, 98
WCSGetElevationDomain, 99
WCSGetTimeDomain, 100
WFSCapabilities, 101, 102, 104, 106, 108, 112
WFSClient, 103, 103
WFSDescribeFeatureType, 105, 106
WFSFeatureType, 102, 105, 107, 108
WFSFeatureTypeElement, 104, 109, 110, 110
WFSGetFeature, 104, 109, 112, 112
WMSCapabilities, 113, 114, 116, 119, 121
WMSClient, 115, 116
WMSGetFeatureInfo, 117, 118, 119, 123
WMSLayer, 114, 117, 120, 121
WPSBoundingBoxData, 124, 124, 127, 129, 140, 158
WPSCapabilities, 125, 125, 128, 135, 140, 142, 143, 157
WPSClient, 126, 127, 128
WPSComplexData, 127, 129, 130, 131, 140, 158
WPSComplexInputDescription, 131, 132, 132
WPSComplexOutputDescription, 133, 134
WPSDescribeProcess, 134, 135
WPSDescriptionParameter, 136, 136
WPSException, 137, 138, 142
WPSEexecute, 139, 139
WPSEexecuteResponse, 141, 141, 142
WPSFormat, 143, 144
WPSInput, 139, 145, 146
WPSInputDescription, 146, 147, 160
WPSLiteralData, 127, 129, 140, 148, 148, 158
WPSLiteralInputDescription, 149, 149, 150
WPSLiteralOutputDescription, 151, 151
WPSOutput, 152, 152, 153
WPSOutputDescription, 154, 154, 160
WPSParameter, 155, 155
WPSProcess, 142, 156, 157
WPSProcessDescription, 126, 129, 140, 157, 158, 159
WPSResponseDocument, 127, 129, 140, 158, 161, 161
WPSStatus, 142, 162, 163