Package 'geometa'

October 28, 2022

Type Package

Title Tools for Reading and Writing ISO/OGC Geographic Metadata

Version 0.7-1

Date 2022-10-27

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

Description Provides facilities to handle reading and writing of geographic metadata defined with OGC/ISO 19115, 11119 and 19110 geographic information metadata standards, and encoded using the ISO 19139 (XML) standard. It includes also a facility to check the validity of ISO 19139 XML encoded metadata.

Depends R (>= 3.3.0)

Imports methods, R6, XML, httr, jsonlite, keyring, readr, crayon

Suggests sf, ncdf4, EML, emld, units, testthat, roxygen2

License MIT + file LICENSE

URL https://github.com/eblondel/geometa/wiki

BugReports https://github.com/eblondel/geometa/issues

LazyLoad yes

RoxygenNote 7.2.1

NeedsCompilation no

Author Emmanuel Blondel [aut, cre] (<https://orcid.org/0000-0002-5870-5762>)

Repository CRAN

Date/Publication 2022-10-27 23:45:13 UTC

R topics documented:

`cacheISOClasses` *cacheISOClasses*

Description

`cacheISOClasses` allows to cache the list of **geometa** classes or extended. This is especially required to fasten the decoding of metadata elements from an XML file. It is called internally by **geometa** the first function `getISOClasses` is called and each time the function `readISO19139` function is called to integrate eventually new classes added by user to extend **geometa** model (case of ISO profiles).

Usage

```
cacheISOClasses()
```

Author(s)

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

Examples

```
cacheISOClasses()
```

`convert_metadata` *convert_metadata*

Description

`convert_metadata` is a tentative generic metadata converter to convert from one source object, represented in a source metadata object model in R (eg eml) to a target metadata object, represented in another target metadata object model (eg **geometa** `ISOMetadata`). This function relies on a list of mapping rules defined to operate from the source metadata object to the target metadata object. This list of mapping rules is provided in a tabular format. A version is embedded in **geometa** and can be returned with `getMappings`.

Usage

```
convert_metadata(obj, from, to, mappings, verbose)
```

Arguments

`obj`	a metadata object given in one of the mapping formats known by **geometa**. The object should be a valid `id` as listed by `getMappingFormats`, supported as source format (`from` is `TRUE`).
`from`	a valid mapping format id (see `getMappingFormats`) that indicates the metadata model / format used for the argument `obj`
`to`	a valid mapping format id (see `getMappingFormats`) to convert to
`mappings`	a `data.frame` giving the reference mapping rules to convert metadata object. This `data.frame` is by default the output of `getMappings`.
`verbose`	print debugging messages. Default is `FALSE`

Value

an metadata object in the model specified as `to` argument

Note

This function is mainly used internally in `as` generic methods to convert from one metadata format to another. It is exported for extension to user custom metadata formats or for debugging purpose. This converter is still experimental.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

`geometa` *Tools for Reading and Writing ISO/OGC Geographic Metadata*

Description

Provides facilities to handle reading and writing of geographic metadata defined with OGC/ISO 19115 and 19110 geographic information metadata standards, and encoded using the ISO 19139 (XML) standard.

Details

Package:	geometa
Type:	Package
Version:	0.6-7
Date:	2022-03-15
License:	MIT
LazyLoad:	yes

Author(s)

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

`geometaLogger` *geometaLogger*

Description

geometaLogger

geometaLogger

Format

`R6Class` object.

Value

Object of `R6Class` for modelling a simple logger

Methods

Public methods:

- `geometaLogger$INFO()`
- `geometaLogger$WARN()`
- `geometaLogger$ERROR()`
- `geometaLogger$new()`
- `geometaLogger$clone()`

Method `INFO()`**:** Logger to report information. Used internally

Usage:

```
geometaLogger$INFO(text)
```

Arguments:

`text` text

Method `WARN()`**:** Logger to report warnings Used internally

Usage:

```
geometaLogger$WARN(text)
```

Arguments:

`text` text

Method `ERROR()`**:** Logger to report errors Used internally

Usage:

```
geometaLogger$ERROR(text)
```

Arguments:

`text` text

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

Usage:

```
geometaLogger$new()
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Logger class used internally by geometa

`geometa_coverage` *geometa_coverage*

Description

`geometa_coverage` is a function to report coverage of ISO/OGC standard classes in package **geometa**. The function will inspect all classes of the ISO/OGC standards and will scan if **geometa** supports it.

Usage

```
geometa_coverage()
```

Value

an object of class `data.frame`

Note

This function is used as Quality Assurance indicator to assess the percentage of completeness of ISO/OGC standards in **geometa**.

Author(s)

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

Examples

```
  cov <- geometa_coverage()
```

`getClassesInheriting` *getClassesInheriting*

Description

get the list of classes inheriting a given super class provided by its name

Usage

```
getClassesInheriting(classname, extended, pretty)
```

Arguments

`classname`	the name of the superclass for which inheriting sub-classes have to be listed
`extended`	whether we want to look at user namespace for third-party sub-classes
`pretty`	prettify the output as `data.frame`

Examples

```
getClassesInheriting("ISAbstractObject")
```

`getGeometaOption` *getGeometaOption*

Description

`getGeometaOption` allows to get an option from **geometa**

Usage

```
getGeometaOption(option)
```

Arguments

`option`	the name of the option

Value

the option

Author(s)

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

Examples

```
getGeometaOption("schemaBaseUrl")
```

`getIANAMimeTypes` *getIANAMimeTypes*

Description

getIANAMimeTypes

Usage

```
getIANAMimeTypes()
```

`getISOClasses` *getISOClasses*

Description

get the list of cached ISO classes

Usage

```
getISOClasses()
```

Author(s)

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

Examples

```
  getISOClasses()
```

`getISOCodelist` *getISOCodelist*

Description

`getISOCodelist` allows to get a registered ISO codelist by id registered in **geometa**

Usage

```
getISOCodelist(id)
```

Arguments

`id`	identifier of the codelist

Author(s)

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

Examples

```
  getISOCodelist(id = "LanguageCode")
```

`getISOCodelists` *getISOCodelists*

Description

`getISOCodelists` allows to get the list of ISO codelists registered in **geometa**, their description and XML definition. The object returned is of class "data.frame"

Usage

```
getISOCodelists()
```

Author(s)

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

Examples

```
getISOCodelists()
```

`getISOInternalCodelists` *getISOInternalCodelists*

Description

`getISOInternalCodelists` allows to get the list of ISO codelists registered in **geometa**

Usage

```
getISOInternalCodelists()
```

Author(s)

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

Examples

```
getISOInternalCodelists()
```

getISOMetadataNamespace
getISOMetadataNamespace

Description

`getISOMetadataNamespace` gets a namespace given its id

Usage

```
getISOMetadataNamespace(id)
```

Arguments

`id`	namespace prefix

Author(s)

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

Examples

```
getISOMetadataNamespace("GMD")
```

getISOMetadataNamespaces
getISOMetadataNamespaces

Description

`getISOMetadataNamespaces` gets the list of namespaces registered

Usage

```
getISOMetadataNamespaces()
```

Author(s)

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

Examples

```
getISOMetadataNamespaces()
```

`getISOMetadataSchemas` *getISOMetadataSchemas*

Description

`getISOMetadataSchemas` gets the schemas registered in **geometa**

Usage

```
getISOMetadataSchemas()
```

Author(s)

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

Examples

```
  getISOMetadataSchemas()
```

`getMappingFormats` *getMappingFormats*

Description

`getMappingFormats` gets the mapping formats registered in **geometa**

Usage

```
getMappingFormats(pretty)
```

Arguments

`pretty`	by default `TRUE` to return the list of formats as `data.frame`. Set to `FALSE` to return a list of `pivot_format` objects

Author(s)

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

`getMappings` *getMappings*

Description

List the mappings rules to convert from/to other metadata formats (currently EML/emld objects and NetCDF-CF/ncdf4 objects)

Usage

```
getMappings()
```

Value

a `data.frame` containing the metadata mapping rules

`GMLAbstractCoordinateOperation` *GMLAbstractCoordinateOperation*

Description

GMLAbstractCoordinateOperation

GMLAbstractCoordinateOperation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLAbstractCoordinateOperation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `GMLAbstractCoordinateOperation`

Public fields

`domainOfValidity` domainOfValidity [0..1]: character

`scope` scope [1..*]: character

`operationVersion` operationVersion [0..1]: character

`coordinateOperationAccuracy` coordinateOperationAccuracy [0..1]: ISOPositionalAccuracy

`sourceCRS` sourceCRS [0..1]: subclass of GMLAbstractCRS

`targetCRS` targetCRS [0..1]: subclass of GMLAbstractCRS

Methods

Public methods:

- `GMLAbstractCoordinateOperation$new()`
- `GMLAbstractCoordinateOperation$setDomainOfValidity()`
- `GMLAbstractCoordinateOperation$addScope()`
- `GMLAbstractCoordinateOperation$delScope()`
- `GMLAbstractCoordinateOperation$setVersion()`
- `GMLAbstractCoordinateOperation$addAccuracy()`
- `GMLAbstractCoordinateOperation$delAccuracy()`
- `GMLAbstractCoordinateOperation$setSourceCRS()`
- `GMLAbstractCoordinateOperation$setTargetCRS()`
- `GMLAbstractCoordinateOperation$clone()`

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

Usage:

```
GMLAbstractCoordinateOperation$new(xml = NULL, defaults = list(), id = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of default values

`id` id

Method `setDomainOfValidity()`**:** Set domain of validity

Usage:

```
GMLAbstractCoordinateOperation$setDomainOfValidity(domainOfValidity)
```

Arguments:

`domainOfValidity` domain of validity, object extending ISOExtent class

Method `addScope()`**:** Adds scope

Usage:

```
GMLAbstractCoordinateOperation$addScope(scope)
```

Arguments:

`scope` scope

Returns: `TRUE` if added, `FALSE` otherwise

Method `delScope()`**:** Removes scope

Usage:

```
GMLAbstractCoordinateOperation$delScope(scope)
```

Arguments:

`scope` scope

Returns: `TRUE` if removed, `FALSE` otherwise

Method `setVersion()`**:** Set version

Usage:

`GMLAbstractCoordinateOperation$setVersion(version)`

Arguments:

`version` version

Method `addAccuracy():` Adds accuracy

Usage:

`GMLAbstractCoordinateOperation$addAccuracy(accuracy)`

Arguments:

`accuracy` accuracy, object inheriting class ISOAbstractPositionalAccuracy

Returns: `TRUE` if added, `FALSE` otherwise

Method `delAccuracy():` Removes accuracy

Usage:

`GMLAbstractCoordinateOperation$delAccuracy(accuracy)`

Arguments:

`accuracy` accuracy, object inheriting class ISOAbstractPositionalAccuracy

Returns: `TRUE` if removed, `FALSE` otherwise

Method `setSourceCRS():` Set source CRS

Usage:

`GMLAbstractCoordinateOperation$setSourceCRS(crs)`

Arguments:

`crs` crs, object inheriting class GMLAbstractSingleCRS

Method `setTargetCRS():` Set target CRS

Usage:

`GMLAbstractCoordinateOperation$setTargetCRS(crs)`

Arguments:

`crs` crs, object inheriting class GMLAbstractSingleCRS

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

Usage:

`GMLAbstractCoordinateOperation$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLAbstractCoordinateSystem` *GMLAbstractCoordinateSystem*

Description

GMLAbstractCoordinateSystem

GMLAbstractCoordinateSystem

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLAbstractCoordinateSystem

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `GMLAbstractCoordinateSystem`

Public fields

`axis` axis [1..*]: GMLCoordinateSystemAxis

Methods

Public methods:

- `GMLAbstractCoordinateSystem$new()`
- `GMLAbstractCoordinateSystem$addAxis()`
- `GMLAbstractCoordinateSystem$delAxis()`
- `GMLAbstractCoordinateSystem$clone()`

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

Usage:

```
GMLAbstractCoordinateSystem$new(xml = NULL, defaults = list(), id = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of default values

`id` id

Method `addAxis()`**:** Adds an axis

Usage:

```
GMLAbstractCoordinateSystem$addAxis(axis)
```

Arguments:

`axis` object of class `GMLCoordinateSystemAxis`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delAxis()`**:** Deletes an axis

Usage:

```
GMLAbstractCoordinateSystem$delAxis(axis)
```

Arguments:

`axis` object of class `GMLCoordinateSystemAxis`

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractCoverage` *GMLAbstractCoverage*

Description

GMLAbstractCoverage

GMLAbstractCoverage

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract coverage

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractFeature` -> `GMLAbstractCoverage`

Public fields

`domainSet` domainSet

`rangeSet` rangeSet

Methods

Public methods:

- `GMLAbstractCoverage$new()`
- `GMLAbstractCoverage$setDomainSet()`
- `GMLAbstractCoverage$setRangeSet()`
- `GMLAbstractCoverage$clone()`

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

Usage:

```
GMLAbstractCoverage$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

Method `setDomainSet()`**:** Set domain set

Usage:

```
GMLAbstractCoverage$setDomainSet(domainSet)
```

Arguments:

`domainSet` object inheriting either GMLAbstractGeometry or GMLAbstractTimeObject

Method `setRangeSet()`**:** Set range set (NOT YET IMPLEMENTED)

Usage:

```
GMLAbstractCoverage$setRangeSet()
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Internal binding used with OGC services

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t
OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractCRS` *GMLAbstractCRS*

Description

GMLAbstractCRS

GMLAbstractCRS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLAbstractCRS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `GMLAbstractCRS`

Public fields

scope scope [1..*]: character

Methods

Public methods:

- `GMLAbstractCRS$new()`
- `GMLAbstractCRS$addScope()`
- `GMLAbstractCRS$delScope()`
- `GMLAbstractCRS$clone()`

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

Usage:

```
GMLAbstractCRS$new(xml = NULL, defaults = list(), id = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of default values

`id` id

Method `addScope()`**:** Adds scope

Usage:

```
GMLAbstractCRS$addScope(scope)
```

Arguments:

`scope` scope

Returns: `TRUE` if added, `FALSE` otherwise

Method `delScope()`**:** Removes scope

Usage:

```
GMLAbstractCRS$delScope(scope)
```

Arguments:

`scope` scope

Returns: `TRUE` if removed, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLAbstractCurve` *GMLAbstractCurve*

Description

GMLAbstractCurve

GMLAbstractCurve

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract curve

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractGeometricPrimitive` -> `GMLAbstractCurve`

Methods

Public methods:

- `GMLAbstractCurve$clone()`

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

Usage:

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

Arguments:

deep	Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLAbstractDiscreteCoverage` *GMLAbstractDiscreteCoverage*

Description

GMLAbstractDiscreteCoverage

GMLAbstractDiscreteCoverage

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract discrete coverage

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractFeature` -> `geometa::GMLAbstractCoverage` -> `GMLAbstractDiscreteCoverage`

Public fields

`coverageFunction` coverage function

Methods

Public methods:

- `GMLAbstractDiscreteCoverage$new()`
- `GMLAbstractDiscreteCoverage$setCoverageFunction()`
- `GMLAbstractDiscreteCoverage$clone()`

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

Usage:

```
GMLAbstractDiscreteCoverage$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

Method `setCoverageFunction()`**:** Set coverage function

Usage:

```
GMLAbstractDiscreteCoverage$setCoverageFunction(coverageFunction)
```

Arguments:

`coverageFunction` object of class GMLGridFunction (or`GMLCoverageMappingRule`, not yet supported)

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t
OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractFeature` *GMLAbstractFeature*

Description

GMLAbstractFeature

GMLAbstractFeature

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract feature

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `GMLAbstractFeature`

Public fields

`boundedBy` boundedBy envelope

Methods

Public methods:

- `GMLAbstractFeature$new()`
- `GMLAbstractFeature$setBoundedBy()`
- `GMLAbstractFeature$clone()`

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

Usage:

```
GMLAbstractFeature$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

Method `setBoundedBy()`**:** Sets bounding envelope

Usage:

```
GMLAbstractFeature$setBoundedBy(envelope)
```

Arguments:

`envelope` envelope, object of class GMLEnvelope

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t
OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLAbstractGeneralConversion` *GMLAbstractGeneralConversion*

Description

GMLAbstractGeneralConversion

GMLAbstractGeneralConversion

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLAbstractGeneralConversion

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateOperation` -> `geometa::GMLAbstractSingleOperation` -> `GMLAbstractGeneralConversion`

Methods

Public methods:

- `GMLAbstractGeneralConversion$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

GMLAbstractGeneralDerivedCRS *GMLAbstractGeneralDerivedCRS*

Description

GMLAbstractGeneralDerivedCRS

GMLAbstractGeneralDerivedCRS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLAbstractGeneralDerivedCRS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCRS` -> `geometa::GMLAbstractSingleCRS` -> `GMLAbstractGeneralDerivedCRS`

Public fields

`conversion` conversion [1..1]: GMLConversion

Methods

Public methods:

- `GMLAbstractGeneralDerivedCRS$setConversion()`
- `GMLAbstractGeneralDerivedCRS$clone()`

Method `setConversion()`**:** Set conversion

Usage:

```
GMLAbstractGeneralDerivedCRS$setConversion(conversion)
```

Arguments:

`conversion,` object of class GMLConversion

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractGeneralOperationParameter` *GMLAbstractGeneralOperationParameter*

Description

GMLAbstractGeneralOperationParameter

GMLAbstractGeneralOperationParameter

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLAbstractGeneralOperationParameter

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `GMLAbstractGeneralOperationParameter`

Public fields

`minimumOccurs` minimumOccurs [0..1]: integer

Methods

Public methods:

- `GMLAbstractGeneralOperationParameter$setMinimumOccurs()`
- `GMLAbstractGeneralOperationParameter$clone()`

Method `setMinimumOccurs()`**:** Set minimum occurs

Usage:

```
GMLAbstractGeneralOperationParameter$setMinimumOccurs(minimumOccurs)
```

Arguments:

`minimumOccurs` object of class integer

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractGeneralParameterValue`
GMLAbstractGeneralParameterValue

Description

GMLAbstractGeneralParameterValue

GMLAbstractGeneralParameterValue

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract general ParameterValue

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `GMLAbstractGeneralParameterValue`

Methods

Public methods:

- `GMLAbstractGeneralParameterValue$new()`
- `GMLAbstractGeneralParameterValue$clone()`

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

Usage:

```
GMLAbstractGeneralParameterValue$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list()
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

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

Usage:

`GMLAbstractGeneralParameterValue$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractGeometricAggregate`
GMLAbstractGeometricAggregate

Description

GMLAbstractGeometricAggregate

GMLAbstractGeometricAggregate

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract Geometric Aggregate

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `GMLAbstractGeometricAggregate`

Methods

Public methods:

- `GMLAbstractGeometricAggregate$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractGeometricPrimitive` *GMLAbstractGeometricPrimitive*

Description

GMLAbstractGeometricPrimitive

GMLAbstractGeometricPrimitive

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract Geometric Primitive

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `GMLAbstractGeometricPrimitive`

Methods

Public methods:

- `GMLAbstractGeometricPrimitive$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLAbstractGeometry` *GMLAbstractGeometry*

Description

GMLAbstractGeometry

GMLAbstractGeometry

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract Geometry

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `GMLAbstractGeometry`

Methods

Public methods:

- GMLAbstractGeometry$new()
- GMLAbstractGeometry$clone()

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

Usage:

```
GMLAbstractGeometry$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLAbstractGML` *GMLAbstractGML*

Description

GMLAbstractGML

GMLAbstractGML

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract GML

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `GMLAbstractGML`

Public fields

`metaDataProperty` metaDataProperty [0..*]

`description` description [0..1]

`descriptionReference` descriptionReference [0..1]: character

`identifier` identifier [0..1]: character

`name` name [0..*]: character

Methods

Public methods:

- `GMLAbstractGML$new()`
- `GMLAbstractGML$setDescription()`
- `GMLAbstractGML$setDescriptionReference()`
- `GMLAbstractGML$setIdentifier()`
- `GMLAbstractGML$addName()`
- `GMLAbstractGML$delName()`
- `GMLAbstractGML$clone()`

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

Usage:

```
GMLAbstractGML$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

Method `setDescription()`**:** Set description

Usage:

```
GMLAbstractGML$setDescription(description)
```

Arguments:

`description` description

Method `setDescriptionReference()`**:** Set description reference

Usage:

```
GMLAbstractGML$setDescriptionReference(descriptionReference)
```

Arguments:

`descriptionReference` description reference

Method `setIdentifier()`**:** Set identifier

Usage:

```
GMLAbstractGML$setIdentifier(identifier, codeSpace)
```

Arguments:

`identifier` identifier

`codeSpace` codespace

Method `addName()`**:** Adds name

Usage:

```
GMLAbstractGML$addName(name, codeSpace = NULL)
```

Arguments:

`name` name

`codeSpace` codespace

Returns: `TRUE` if added, `FALSE` otherwise

Method `delName()`**:** Deletes name

Usage:

```
GMLAbstractGML$delName(name, codeSpace = NULL)
```

Arguments:

`name` name

`codeSpace` codespace

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractImplicitGeometry`

GMLAbstractImplicitGeometry

Description

GMLAbstractImplicitGeometry

GMLAbstractImplicitGeometry

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract implicit Geometry

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `GMLAbstractImplicitGeometry`

Methods

Public methods:

- `GMLAbstractImplicitGeometry$new()`
- `GMLAbstractImplicitGeometry$clone()`

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

Usage:

```
GMLAbstractImplicitGeometry$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLAbstractObject` *GMLAbstractObject*

Description

GMLAbstractObject

GMLAbstractObject

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract object

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `GMLAbstractObject`

Methods

Public methods:

- `GMLAbstractObject$new()`
- `GMLAbstractObject$clone()`

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

Usage:

```
GMLAbstractObject$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = FALSE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractReferenceableGrid`

GMLAbstractReferenceableGrid

Description

GMLAbstractReferenceableGrid

GMLAbstractReferenceableGrid

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML grid

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractImplicitGeometry` -> `geometa::GMLGrid` -> `GMLAbstractReferenceableGrid`

Methods

Public methods:

- `GMLAbstractReferenceableGrid$new()`
- `GMLAbstractReferenceableGrid$clone()`

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

Usage:

```
GMLAbstractReferenceableGrid$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

OGC GML 3.3 Schema. http://schemas.opengis.net/gml/3.3/referenceableGrid.xsd

---

`GMLAbstractRing` *GMLAbstractRing*

Description

GMLAbstractRing

GMLAbstractRing

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract ring

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `GMLAbstractRing`

Methods

Public methods:

- GMLAbstractRing$clone()

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractSingleCRS` *GMLAbstractSingleCRS*

Description

GMLAbstractSingleCRS

GMLAbstractSingleCRS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLAbstractSingleCRS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCRS` -> `GMLAbstractSingleCRS`

Methods

Public methods:

- `GMLAbstractSingleCRS$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractSingleOperation` *GMLAbstractSingleOperation*

Description

GMLAbstractSingleOperation

GMLAbstractSingleOperation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLAbstractSingleOperation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateOperation` -> `GMLAbstractSingleOperation`

Methods

Public methods:

- `GMLAbstractSingleOperation$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractSurface` *GMLAbstractSurface*

Description

GMLAbstractSurface

GMLAbstractSurface

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML abstract surface

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractGeometricPrimitive` -> `GMLAbstractSurface`

Methods

Public methods:

- `GMLAbstractSurface$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractTimeGeometricPrimitive`

GMLAbstractTimeGeometricPrimitive

Description

GMLAbstractTimeGeometricPrimitive

GMLAbstractTimeGeometricPrimitive

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO GML abstract temporal primitive

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractTimeObject` -> `geometa::GMLAbstractTimePrimitive` -> `GMLAbstractTimeGeometricPrimitive`

Methods

Public methods:

- `GMLAbstractTimeGeometricPrimitive$new()`
- `GMLAbstractTimeGeometricPrimitive$clone()`

Method `new()`: Initializes object

Usage:

```
GMLAbstractTimeGeometricPrimitive$new(xml = NULL, defaults = list())
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of default values

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`GMLAbstractTimeObject` *GMLAbstractTimeObject*

Description

GMLAbstractTimeObject

GMLAbstractTimeObject

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML AbstractTimeObject

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `GMLAbstractTimeObject`

Methods

Public methods:

- GMLAbstractTimeObject$new()
- GMLAbstractTimeObject$clone()

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

Usage:

```
GMLAbstractTimeObject$new(xml = NULL, defaults = list())
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of default values

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAbstractTimePrimitive` *GMLAbstractTimePrimitive*

Description

GMLAbstractTimePrimitive

GMLAbstractTimePrimitive

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML AbstractTimePrimitive

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractTimeObject` -> `GMLAbstractTimePrimitive`

Public fields

`relatedTime` relatedTime

Methods

Public methods:

- `GMLAbstractTimePrimitive$new()`
- `GMLAbstractTimePrimitive$addRelatedTime()`
- `GMLAbstractTimePrimitive$delRelatedTime()`
- `GMLAbstractTimePrimitive$clone()`

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

Usage:

```
GMLAbstractTimePrimitive$new(xml = NULL, defaults = list())
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of default values

Method `addRelatedTime()`**:** Adds related time

Usage:

```
GMLAbstractTimePrimitive$addRelatedTime(time)
```

Arguments:

`time` object of class GMLTimeInstant, GMLTimePeriod. (`GMLTimeNode` or `GMLTimeEdge` are not yet supported)

Returns: `TRUE` if added, `FALSE` otherwise

Method `delRelatedTime()`**:** Deletes related time

Usage:

```
GMLAbstractTimePrimitive$delRelatedTime(time)
```

Arguments:

`time` object of class GMLTimeInstant, GMLTimePeriod. (`GMLTimeNode` or `GMLTimeEdge` are not yet supported)

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLAffineCS` *GMLAffineCS*

Description

GMLAffineCS

GMLAffineCS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLAffineCS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateSystem` -> `GMLAffineCS`

Methods

Public methods:

- `GMLAffineCS$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLBaseUnit` *GMLBaseUnit*

Description

GMLBaseUnit

GMLBaseUnit

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML base unit

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLUnitDefinition` -> `GMLBaseUnit`

Public fields

`unitsSystem` unitsSystem [1..1]: character

Methods

Public methods:

- `GMLBaseUnit$new()`
- `GMLBaseUnit$setUnitsSystem()`
- `GMLBaseUnit$clone()`

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

Usage:

`GMLBaseUnit$new(xml = NULL, defaults = list(), id = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of default values

`id` id

Method `setUnitsSystem()`**:** Set unit system

Usage:

`GMLBaseUnit$setUnitsSystem(unitsSystem)`

Arguments:

`unitsSystem` units system

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

Usage:

`GMLBaseUnit$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

Examples

```
gml <- GMLBaseUnit$new()
gml$setDescriptionReference("someref")
gml$setIdentifier("identifier", "codespace")
gml$addName("name1", "codespace")
gml$addName("name2", "codespace")
gml$setQuantityTypeReference("someref")
gml$setCatalogSymbol("symbol")
gml$setUnitsSystem("somelink")
```

`GMLCartesianCS` *GMLCartesianCS*

Description

GMLCartesianCS

GMLCartesianCS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLCartesianCS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateSystem` -> `GMLCartesianCS`

Methods

Public methods:

- `GMLCartesianCS$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLCodeType` *GMLCodeType*

Description

GMLCodeType

GMLCodeType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling a GML code type

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `GMLCodeType`

Public fields

`value` value

`attrs` attributes

Methods

Public methods:

- `GMLCodeType$new()`
- `GMLCodeType$clone()`

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

Usage:

```
GMLCodeType$new(xml = NULL, value = NULL, codeSpace = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`codeSpace` code space

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLCompoundCRS` *GMLCompoundCRS*

Description

GMLCompoundCRS

GMLCompoundCRS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLCompoundCRS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCRS` -> `GMLCompoundCRS`

Public fields

`componentReferenceSystem` componentReferenceSystem [2..*]: instance of AbstractSingleCRS

Methods

Public methods:

- `GMLCompoundCRS$new()`
- `GMLCompoundCRS$addComponentReferenceSystem()`
- `GMLCompoundCRS$delComponentReferenceSystem()`
- `GMLCompoundCRS$clone()`

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

Usage:

```
GMLCompoundCRS$new(xml = NULL, defaults = list(), id = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` default values

`id` id

Method `addComponentReferenceSystem()`**:** Adds component reference system

Usage:

```
GMLCompoundCRS$addComponentReferenceSystem(referenceSystem)
```

Arguments:

`referenceSystem` referenceSystem, object of class GMLAbstractSingleCRS

Returns: `TRUE` if added, `FALSE` otherwise

Method `delComponentReferenceSystem()`**:** Deletes component reference system

Usage:

```
GMLCompoundCRS$delComponentReferenceSystem(referenceSystem)
```

Arguments:

`referenceSystem` referenceSystem, object of class GMLAbstractSingleCRS

Returns: `TRUE` if delete, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLConventionalUnit` *GMLConventionalUnit*

Description

GMLConventionalUnit

GMLConventionalUnit

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML derived unit

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLUnitDefinition` -> `GMLConventionalUnit`

Public fields

`conversionToPreferredUnit` conversionToPreferredUnit [1..1]: character/integer

`roughConversionToPreferredUnit` roughConversionToPreferredUnit [1..1]: character/integer

`derivationUnitTerm` derivationUnitTerm [1..*]: character

Methods

Public methods:

- `GMLConventionalUnit$new()`
- `GMLConventionalUnit$addDerivationUnitTerm()`
- `GMLConventionalUnit$delDerivationUnitTerm()`
- `GMLConventionalUnit$setConversionToPreferredUnit()`
- `GMLConventionalUnit$clone()`

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

Usage:

```
GMLConventionalUnit$new(xml = NULL, defaults = list(), id = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` default values

`id` id

Method `addDerivationUnitTerm()`**:** Adds a derivation unit term, made of a uom reference, and an exponent which can be negative/positive but not equal to zero.

Usage:

```
GMLConventionalUnit$addDerivationUnitTerm(uom, exponent)
```

Arguments:

`uom` unit of measure reference

`exponent` exponent

Returns: `TRUE` if added, `FALSE` `otherwise`

Method `delDerivationUnitTerm()`**:** Deletes a derivation unit term

Usage:

```
GMLConventionalUnit$delDerivationUnitTerm(uom, exponent)
```

Arguments:

`uom` unit of measure reference

`exponent` exponent

Returns: `TRUE` if deleted, `FALSE` `otherwise`

Method `setConversionToPreferredUnit()`**:** Sets the conversion to preferred unit.

Usage:

```
GMLConventionalUnit$setConversionToPreferredUnit(uom, factor, rough = FALSE)
```

Arguments:

`uom` unit of measure reference

`factor` factor

`rough` rough . Defaut is `FALSE`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

Examples

```
gml <- GMLConventionalUnit$new()
gml$setDescriptionReference("someref")
gml$setIdentifier("identifier", "codespace")
gml$addName("name1", "codespace")
gml$addName("name2", "codespace")
gml$setQuantityTypeReference("someref")
gml$setCatalogSymbol("symbol")
gml$addDerivationUnitTerm("uomId", 2L)
gml$setConversionToPreferredUnit("uomId", 2L)
```

`GMLConversion` *GMLConversion*

Description

GMLConversion

GMLConversion

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLConversion

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateOperation` -> `geometa::GMLAbstractSingleOperation` -> `geometa::GMLAbstractGeneralConversion` -> `GMLConversion`

Public fields

`method` method [1..1]: GMLOperationMethod

`parameterValue` parameterValue [0..*]: GMLParameterValue

Methods

Public methods:

- `GMLConversion$setMethod()`
- `GMLConversion$addParameterValue()`
- `GMLConversion$delParameterValue()`
- `GMLConversion$clone()`

Method `setMethod():` Set method

Usage:

```
GMLConversion$setMethod(method)
```

Arguments:

`method` method, object of class GMLOperationMethod

Method `addParameterValue():` Adds parameter value

Usage:

```
GMLConversion$addParameterValue(paramValue)
```

Arguments:

`paramValue` parameter value, object class inheriting GMLAbstractGeneralParameterValue

Returns: `TRUE` if added, `FALSE` otherwise

Method `delParameterValue():` Deletes parameter value

Usage:

```
GMLConversion$delParameterValue(paramValue)
```

Arguments:

`paramValue` parameter value, object class inheriting GMLAbstractGeneralParameterValue

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLCoordinateSystemAxis` *GMLCoordinateSystemAxis*

Description

GMLCoordinateSystemAxis

GMLCoordinateSystemAxis

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLCoordinateSystemAxis

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `GMLCoordinateSystemAxis`

Public fields

`axisAbbrev` axisAbbrev [1..1]: character

`axisDirection` axisDirection [1..1]: character (with codeSpace)

`minimumValue` minimumValue [0..1]: double

`maximumValue` maximumValue [0..1]: double

`rangeMeaning` rangeMeaning [0..1]: character (with codeSpace)

Methods

Public methods:

- `GMLCoordinateSystemAxis$new()`
- `GMLCoordinateSystemAxis$setAbbrev()`
- `GMLCoordinateSystemAxis$setDirection()`
- `GMLCoordinateSystemAxis$setMinimumValue()`
- `GMLCoordinateSystemAxis$setMaximumValue()`
- `GMLCoordinateSystemAxis$setRangeMeaning()`
- `GMLCoordinateSystemAxis$clone()`

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

Usage:

```
GMLCoordinateSystemAxis$new(xml = NULL, defaults = list(), id = NULL, uom = NA)
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of default values

`id` id

`uom` unit of measure

Method `setAbbrev():` Set Abbrev

Usage:

```
GMLCoordinateSystemAxis$setAbbrev(abbrev)
```

Arguments:

`abbrev` abbrev

Method `setDirection():` Set description

Usage:

```
GMLCoordinateSystemAxis$setDirection(direction, codeSpace = NULL)
```

Arguments:

`direction` direction

`codeSpace` code space

Method `setMinimumValue():` Set minimum value

Usage:

```
GMLCoordinateSystemAxis$setMinimumValue(value)
```

Arguments:

`value` value

Method `setMaximumValue():` Set maxium value

Usage:

```
GMLCoordinateSystemAxis$setMaximumValue(value)
```

Arguments:

`value` value

Method `setRangeMeaning():` Set range meaning

Usage:

```
GMLCoordinateSystemAxis$setRangeMeaning(meaning, codeSpace = NULL)
```

Arguments:

`meaning` meaning

`codeSpace` code space

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLCOVAbstractCoverage` *GMLCOVAbstractCoverage*

Description

GMLCOVAbstractCoverage

GMLCOVAbstractCoverage

Format

`R6Class` object.

Value

Object of `R6Class` for modelling a GMLCOV Abstract Coverage

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractFeature` -> `geometa::GMLAbstractCoverage` -> `GMLCOVAbstractCoverage`

Public fields

`coverageFunction` coverage function

`rangeType` range type

`metadata` metadata

Methods

Public methods:

- `GMLCOVAbstractCoverage$new()`
- `GMLCOVAbstractCoverage$clone()`

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

Usage:

```
GMLCOVAbstractCoverage$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

GML 3.2.1 Application Schema for Coverages http://www.opengis.net/gmlcov/1.0

`GMLCOVExtension` *GMLCOVExtension*

Description

GMLCOVExtension

GMLCOVExtension

Format

`R6Class` object.

Value

Object of `R6Class` for modelling a GMLCOV Extension

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `GMLCOVExtension`

Public fields

`anyElement` anyElement

Methods

Public methods:

- `GMLCOVExtension$new()`
- `GMLCOVExtension$clone()`

Method `new()`: Initializes object

Usage:

```
GMLCOVExtension$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Internal binding for OGC services

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

GML 3.2.1 Application Schema for Coverages http://www.opengis.net/gmlcov/1.0

`GMLCylindricalCS` *GMLCylindricalCS*

Description

GMLCylindricalCS

GMLCylindricalCS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLCylindricalCS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateSystem` -> `GMLCylindricalCS`

Methods

Public methods:

- `GMLCylindricalCS$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLDefinition` *GMLDefinition*

Description

GMLDefinition

GMLDefinition

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML definition

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `GMLDefinition`

Public fields

`remarks` remarks [0..*]: character

Methods

Public methods:

- `GMLDefinition$new()`
- `GMLDefinition$addRemark()`
- `GMLDefinition$delRemark()`
- `GMLDefinition$clone()`

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

Usage:

```
GMLDefinition$new(xml = NULL, defaults = list())
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` default values

Method `addRemark()`**:** Adds remark

Usage:

```
GMLDefinition$addRemark(remark)
```

Arguments:

`remark` remark

Returns: `TRUE` if added, `FALSE` otherwise

Method `delRemark()`**:** Deletes remark

Usage:

`GMLDefinition$delRemark(remark)`

Arguments:

`remark` remark

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`GMLDefinition$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

Examples

```
gml <- GMLDefinition$new()
gml$setDescriptionReference("someref")
gml$setIdentifier("identifier", "codespace")
gml$addName("name1", "codespace")
gml$addName("name2", "codespace")
```

`GMLDerivedCRS` *GMLDerivedCRS*

Description

GMLDerivedCRS

GMLDerivedCRS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLDerivedCRS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCRS` -> `geometa::GMLAbstractSingleCRS` -> `geometa::GMLAbstractGeneralDerivedCRS` -> `GMLDerivedCRS`

Public fields

`baseCRS` baseCRS [1..1]: inherited from GMLAbstractSingleCRS

`derivedCRSType` derivedCRSType [1..1]: character

`coordinateSystem` coordinateSystem [1..1]: inherited from GMLAbstractCoordinateSystem

Methods

Public methods:

- `GMLDerivedCRS$setBaseCRS()`
- `GMLDerivedCRS$setDerivedCRSType()`
- `GMLDerivedCRS$setCoordinateSystem()`
- `GMLDerivedCRS$clone()`

Method `setBaseCRS()`**:** Set base CRS

Usage:

```
GMLDerivedCRS$setBaseCRS(crs)
```

Arguments:

`crs` object inheriting class GMLAbstractSingleCRS

Method `setDerivedCRSType()`**:** Set derived CRS type

Usage:

```
GMLDerivedCRS$setDerivedCRSType(type, codeSpace = NULL)
```

Arguments:

`type` type

`codeSpace` code space

Method `setCoordinateSystem()`**:** set coordinate system

Usage:

```
GMLDerivedCRS$setCoordinateSystem(cs)
```

Arguments:

`cs` cs, object inheriting class GMLAbstractCoordinateSystem

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLDerivedUnit` *GMLDerivedUnit*

Description

GMLDerivedUnit

GMLDerivedUnit

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML derived unit

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLUnitDefinition` -> `GMLDerivedUnit`

Public fields

`derivationUnitTerm` derivationUnitTerm [1..*]: character

Methods

Public methods:

- `GMLDerivedUnit$new()`
- `GMLDerivedUnit$addDerivationUnitTerm()`
- `GMLDerivedUnit$delDerivationUnitTerm()`
- `GMLDerivedUnit$clone()`

Method `new()`: Initializes object

Usage:

```
GMLDerivedUnit$new(xml = NULL, defaults = list(), id = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` default values

`id` id

Method `addDerivationUnitTerm():` Adds a derivation unit term, made of a uom reference, and an exponent which can be negative/positive but not equal to zero.

Usage:

```
GMLDerivedUnit$addDerivationUnitTerm(uom, exponent)
```

Arguments:

`uom` unit of measure reference

`exponent` exponent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDerivationUnitTerm():` Deletes a derivation unit term.

Usage:

```
GMLDerivedUnit$delDerivationUnitTerm(uom, exponent)
```

Arguments:

`uom` unit of measure reference

`exponent` exponent

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

Examples

```
gml <- GMLDerivedUnit$new()
gml$setDescriptionReference("someref")
gml$setIdentifier("identifier", "codespace")
gml$addName("name2", "codespace")
gml$setQuantityTypeReference("someref")
gml$setCatalogSymbol("symbol")
gml$addDerivationUnitTerm("uomId", 2L)
```

GMLElement *GMLElement*

Description

GMLElement

GMLElement

Format

R6Class object.

Value

Object of R6Class for modelling an GML element

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::GMLAbstractObject -> GMLElement

Methods

Public methods:

- GMLElement$new()
- GMLElement$decode()
- GMLElement$clone()

Method `new()`: Initializes object

Usage:

```
GMLElement$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  xmlNamespacePrefix = "GML"
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element

`attrs` attrs

`defaults` default values

`xmlNamespacePrefix` xmlNamespacePrefix Default is ’GML’

Method `decode()`: Decodes the XML

Usage:

`GMLElement$decode(xml)`

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

`GMLElement$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`GMLEllipsoidalCS` *GMLEllipsoidalCS*

Description

GMLEllipsoidalCS

GMLEllipsoidalCS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLEllipsoidalCS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateSystem` -> `GMLEllipsoidalCS`

Methods

Public methods:

- `GMLEllipsoidalCS$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLEnvelope` *GMLEnvelope*

Description

GMLEnvelope

GMLEnvelope

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML envelope

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `GMLEnvelope`

Public fields

`lowerCorner` lower corner

`upperCorner` upper corner

Methods

Public methods:

- `GMLEnvelope$new()`
- `GMLEnvelope$decode()`
- `GMLEnvelope$clone()`

Method `new()`**:** Initializes a GML envelope. The argument 'bbox' should be a matrix of dim 2,2 giving the x/y min/max values of a bouding box, as returned by `bbox` function in package **sp**.

Usage:

```
GMLEnvelope$new(
  xml = NULL,
  element = NULL,
  bbox,
  srsName = NULL,
  srsDimension = NULL,
  axisLabels = NULL,
  uomLabels = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element

`bbox` object of class matrix

`srsName` SRS name

`srsDimension` SRS dimension

`axisLabels` axis labels

`uomLabels` uom labels

Method `decode()`**:** Decodes an XML representation

Usage:

```
GMLEnvelope$decode(xml)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLEnvelopeWithTimePeriod` *GMLEnvelopeWithTimePeriod*

Description

GMLEnvelopeWithTimePeriod

GMLEnvelopeWithTimePeriod

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML envelope with time period

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLEnvelope` -> `GMLEnvelopeWithTimePeriod`

Public fields

`beginPosition` begin position

`endPosition` end position

Methods

Public methods:

- `GMLEnvelopeWithTimePeriod$new()`
- `GMLEnvelopeWithTimePeriod$decode()`
- `GMLEnvelopeWithTimePeriod$setBeginPosition()`
- `GMLEnvelopeWithTimePeriod$setEndPosition()`
- `GMLEnvelopeWithTimePeriod$clone()`

Method `new()`**:** Initializes a GML envelope with time period. The argument 'bbox' should be a matrix of dim 2,2 giving the x/y min/max values of a bouding box, as returned by `bbox` function in package **sp**.

Usage:

```
GMLEnvelopeWithTimePeriod$new(
  xml = NULL,
  element = NULL,
  bbox,
  beginPosition,
  endPosition,
  srsName = NULL,
  srsDimension = NULL,
  axisLabels = NULL,
  uomLabels = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element

`bbox` object of class matrix

`beginPosition` begin position, object of class Date or POSIXct-class

`endPosition` end position, object of class Date or POSIXct-class

`srsName` SRS name

`srsDimension` SRS dimension

`axisLabels` axis labels

`uomLabels` uom labels

Method `decode()`**:** Decodes an XML representation

Usage:

```
GMLEnvelopeWithTimePeriod$decode(xml)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setBeginPosition()`**:** Set begin position

Usage:

```
GMLEnvelopeWithTimePeriod$setBeginPosition(beginPosition)
```

Arguments:

`beginPosition` object of class Date or POSIXct-class

Method `setEndPosition()`: Set end position

Usage:

```
GMLEnvelopeWithTimePeriod$setEndPosition(endPosition)
```

Arguments:

`endPosition` object of class Date or POSIXct-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLGeneralGridAxis` *GMLGeneralGridAxis*

Description

GMLGeneralGridAxis

GMLGeneralGridAxis

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML GeneralGridAxis

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `GMLGeneralGridAxis`

Public fields

`offsetVector` offset vector

`coefficients` coefficients

`gridAxesSpanned` grid axes spanned

`sequenceRule` sequence rule

Methods

Public methods:

- `GMLGeneralGridAxis$new()`
- `GMLGeneralGridAxis$setOffsetVector()`
- `GMLGeneralGridAxis$setCoefficients()`
- `GMLGeneralGridAxis$setGridAxesSpanned()`
- `GMLGeneralGridAxis$setSequenceRule()`
- `GMLGeneralGridAxis$clone()`

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

Usage:

```
GMLGeneralGridAxis$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setOffsetVector()`**:** Set offset vector

Usage:

```
GMLGeneralGridAxis$setOffsetVector(offsetVector)
```

Arguments:

`offsetVector` offset vector object of class vector

Method `setCoefficients()`**:** Set coefficients

Usage:

```
GMLGeneralGridAxis$setCoefficients(coefficients)
```

Arguments:

`coefficients` coefficients object of class vector

Method `setGridAxesSpanned()`**:** Set grid axes spanned

Usage:

```
GMLGeneralGridAxis$setGridAxesSpanned(spanned)
```

Arguments:

`spanned` spanned

Method `setSequenceRule()`**:** Set sequence rule

Usage:

`GMLGeneralGridAxis$setSequenceRule(sequenceRule)`

Arguments:

`sequenceRule` sequence rule

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

Usage:

`GMLGeneralGridAxis$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

OGC GML 3.3 Schema. http://schemas.opengis.net/gml/3.3/referenceableGrid.xsd

---

`GMLGeodeticCRS` *GMLGeodeticCRS*

Description

GMLGeodeticCRS

GMLGeodeticCRS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLGeodeticCRS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCRS` -> `GMLGeodeticCRS`

Public fields

`ellipsoidalCS` ellipsoidalCS [1..1]: GMLEllipsoidalCS

`cartesianCS` cartesianCS [1..1]: GMLCartesianCS

`sphericalCS` sphericalCS [1..1]: GMLSphericalCS

`geodeticDatum` geodeticDatum [1..1]: GMLGeodeticDatum

Methods

Public methods:

- `GMLGeodeticCRS$setEllipsoidalCS()`
- `GMLGeodeticCRS$setCartesianCS()`
- `GMLGeodeticCRS$setSphericalCS()`
- `GMLGeodeticCRS$setGeodeticDatum()`
- `GMLGeodeticCRS$clone()`

Method `setEllipsoidalCS()`**:** Set ellipsoidal CS

Usage:

```
GMLGeodeticCRS$setEllipsoidalCS(cs)
```

Arguments:

`cs` cs, object of class GMLEllipsoidalCS

Method `setCartesianCS()`**:** Set cartesian CS

Usage:

```
GMLGeodeticCRS$setCartesianCS(cs)
```

Arguments:

`cs` cs, object of class GMLCartesianCS

Method `setSphericalCS()`**:** Set spherical CS

Usage:

```
GMLGeodeticCRS$setSphericalCS(cs)
```

Arguments:

`cs` cs, object of class GMLSphericalCS

Method `setGeodeticDatum()`**:** Set geodetic datum. Currently not supported

Usage:

```
GMLGeodeticCRS$setGeodeticDatum(datum)
```

Arguments:

`datum` object of class `GMLGeodeticDatum`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLGrid` *GMLGrid*

Description

GMLGrid

GMLGrid

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML grid

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractImplicitGeometry` -> `GMLGrid`

Public fields

`limits` limits

`axisLabels` axis labels

`axisName` axis name

Methods

Public methods:

- `GMLGrid$new()`
- `GMLGrid$setGridEnvelope()`
- `GMLGrid$setAxisLabels()`
- `GMLGrid$addAxisName()`
- `GMLGrid$delAxisName()`
- `GMLGrid$clone()`

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

Usage:

```
GMLGrid$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

Method `setGridEnvelope()`**:** Set grid envelope

Usage:

```
GMLGrid$setGridEnvelope(m)
```

Arguments:

`m` object of class matrix

Method `setAxisLabels()`**:** Set axis labels

Usage:

```
GMLGrid$setAxisLabels(labels)
```

Arguments:

`labels` labels

Method `addAxisName()`**:** Adds axis name

Usage:

```
GMLGrid$addAxisName(axisName)
```

Arguments:

`axisName` axis name

Returns: `TRUE` if added, `FALSE` otherwise

Method `delAxisName()`**:** Deletes axis name

Usage:

```
GMLGrid$delAxisName(axisName)
```

Arguments:

`axisName` axis name

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

```
GMLGrid$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>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLGridCoverage` *GMLGridCoverage*

Description

GMLGridCoverage

GMLGridCoverage

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML grid coverage

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractFeature` -> `geometa::GMLAbstractCoverage` -> `geometa::GMLAbstractDiscreteCoverage` -> `GMLGridCoverage`

Methods

Public methods:

- `GMLGridCoverage$new()`
- `GMLGridCoverage$clone()`

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

Usage:

```
GMLGridCoverage$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

`GMLGridCoverage$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t
OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLGridEnvelope` *GMLGridEnvelope*

Description

GMLGridEnvelope

GMLGridEnvelope

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML grid envelope

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `GMLGridEnvelope`

Public fields

`low` low value [`matrix`]

`high` high value [`matrix`]

Methods

Public methods:

- `GMLGridEnvelope$new()`
- `GMLGridEnvelope$clone()`

Method `new()`**:** This method is used to instantiate a GML envelope. The argument 'bbox' should be a matrix of dim 2,2 giving the x/y min/max values of a bouding box, as returned by bbox function in package **sp**

Usage:

```
GMLGridEnvelope$new(xml = NULL, bbox)
```

Arguments:

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

`bbox` object of class `matrix`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

GMLGridFunction *GMLGridFunction*

Description

GMLGridFunction

GMLGridFunction

Format

R6Class object.

Value

Object of R6Class for modelling an GML grid function

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::GMLAbstractObject -> GMLGridFunction

Public fields

sequenceRule sequence rule

startPoint start point

Methods

Public methods:

- GMLGridFunction$new()
- GMLGridFunction$setSequenceRule()
- GMLGridFunction$setStartPoint()
- GMLGridFunction$clone()

Method new()**:** Initializes object

Usage:

```
GMLGridFunction$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

xml object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

Method `setSequenceRule():` Set sequence rule

Usage:

`GMLGridFunction$setSequenceRule(sequenceRule)`

Arguments:

`sequenceRule` sequence rule, a value among: Linear,Boustrophedonic, Cantor-diagonal,Spiral,Morton,Hilbert

Method `setStartPoint():` Set start point

Usage:

`GMLGridFunction$setStartPoint(x, y)`

Arguments:

`x` x

`y` y

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

Usage:

`GMLGridFunction$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>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t
OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

GMLLinearCS *GMLLinearCS*

Description

GMLLinearCS

GMLLinearCS

Format

R6Class object.

Value

Object of R6Class for modelling an GMLLinearCS

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::GMLAbstractObject -> geometa::GMLAbstractGML -> geometa::GMLDefinition -> geometa::GMLAbstractCoordinateSystem -> GMLLinearCS

Methods

Public methods:

- GMLLinearCS$clone()

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

Usage:

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

Arguments:

deep	Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLLinearRing` *GMLLinearRing*

Description

GMLLinearRing

GMLLinearRing

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML LinearRing

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractRing` -> `GMLLinearRing`

Public fields

`attrs` gml attributes

`posList` list of positions

Methods

Public methods:

- `GMLLinearRing$new()`
- `GMLLinearRing$clone()`

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

Usage:

```
GMLLinearRing$new(xml = NULL, m)
```

Arguments:

`xml` object of class XMLInternalNode-class

`m` simple object of class matrix

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLLineString` *GMLLineString*

Description

GMLLineString

GMLLineString

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML linestring

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractGeometricPrimitive` -> `geometa::GMLAbstractCurve` -> `GMLLineString`

Public fields

`posList` list of positions

Methods

Public methods:

- `GMLLineString$new()`
- `GMLLineString$clone()`

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

Usage:

`GMLLineString$new(xml = NULL, sfg)`

Arguments:

`xml` object of class XMLInternalNode-class

`sfg` simple feature geometry resulting from **sf**

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

Usage:

`GMLLineString$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLMultiCurve` *GMLMultiCurve*

Description

GMLMultiCurve

GMLMultiCurve

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML multicurve

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractGeometricAggregate` -> `GMLMultiCurve`

Public fields

`attrs` gml attributes

`curveMember` curve members

Methods

Public methods:

- `GMLMultiCurve$new()`
- `GMLMultiCurve$addCurveMember()`
- `GMLMultiCurve$delCurveMember()`
- `GMLMultiCurve$clone()`

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

Usage:

```
GMLMultiCurve$new(xml = NULL, sfg = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`sfg` simple feature geometry resulting from **sf**

Method `addCurveMember()`**:** Adds curve member

Usage:

```
GMLMultiCurve$addCurveMember(curve)
```

Arguments:

`curve` curve object of class inheriting GMLAbstractCurve

Returns: `TRUE` if added, `FALSE` otherwise

Method `delCurveMember()`**:** Deletes curve member

Usage:

```
GMLMultiCurve$delCurveMember(curve)
```

Arguments:

`curve` curve object of class inheriting GMLAbstractCurve

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLMultiCurveCoverage` *GMLMultiCurveCoverage*

Description

GMLMultiCurveCoverage

GMLMultiCurveCoverage

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML multicurve coverage

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractFeature` -> `geometa::GMLAbstractCoverage` -> `geometa::GMLAbstractDiscreteCoverage` -> `GMLMultiCurveCoverage`

Methods

Public methods:

- `GMLMultiCurveCoverage$new()`
- `GMLMultiCurveCoverage$clone()`

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

Usage:

```
GMLMultiCurveCoverage$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

```
GMLMultiCurveCoverage$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>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t
OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLMultiPoint` *GMLMultiPoint*

Description

GMLMultiPoint

GMLMultiPoint

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML multipoint

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractGeometricAggregate` -> `GMLMultiPoint`

Public fields

`pointMember` point members

Methods

Public methods:

- `GMLMultiPoint$new()`
- `GMLMultiPoint$addPointMember()`
- `GMLMultiPoint$delPointMember()`
- `GMLMultiPoint$clone()`

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

Usage:

```
GMLMultiPoint$new(xml = NULL, sfg = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`sfg` simple feature geometry resulting from **sf**

Method `addPointMember()`**:** Adds point member

Usage:

```
GMLMultiPoint$addPointMember(point)
```

Arguments:

`point` point object of class GMLPoint

Returns: `TRUE` if added, `FALSE` otherwise

Method `delPointMember()`**:** Deletes point member

Usage:

```
GMLMultiPoint$delPointMember(point)
```

Arguments:

`point` point object of class GMLPoint

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLMultiPointCoverage` *GMLMultiPointCoverage*

Description

GMLMultiPointCoverage

GMLMultiPointCoverage

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML multipoint coverage

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractFeature` -> `geometa::GMLAbstractCoverage` -> `geometa::GMLAbstractDiscreteCoverage` -> `GMLMultiPointCoverage`

Methods

Public methods:

- `GMLMultiPointCoverage$new()`
- `GMLMultiPointCoverage$clone()`

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

Usage:

```
GMLMultiPointCoverage$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t
OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLMultiSolidCoverage` *GMLMultiSolidCoverage*

Description

GMLMultiSolidCoverage

GMLMultiSolidCoverage

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML multisolid coverage

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractFeature` -> `geometa::GMLAbstractCoverage` -> `geometa::GMLAbstractDiscreteCoverage` -> `GMLMultiSolidCoverage`

Methods

Public methods:

- GMLMultiSolidCoverage$new()
- GMLMultiSolidCoverage$clone()

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

Usage:

```
GMLMultiSolidCoverage$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t
OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLMultiSurface` *GMLMultiSurface*

Description

GMLMultiSurface

GMLMultiSurface

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML multisurface

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractGeometricAggregate` -> `GMLMultiSurface`

Public fields

`attrs` gml attributes

`surfaceMember` surface members

Methods

Public methods:

- `GMLMultiSurface$new()`
- `GMLMultiSurface$addSurfaceMember()`
- `GMLMultiSurface$delSurfaceMember()`
- `GMLMultiSurface$clone()`

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

Usage:

```
GMLMultiSurface$new(xml = NULL, sfg = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`sfg` simple feature geometry resulting from **sf**

Method `addSurfaceMember()`**:** Adds surface member

Usage:

```
GMLMultiSurface$addSurfaceMember(surface)
```

Arguments:

`surface` surface object of class inheriting GMLAbstractSurface

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSurfaceMember()`**:** Deletes surface member

Usage:

```
GMLMultiSurface$delSurfaceMember(surface)
```

Arguments:

`surface` surface object of class inheriting GMLAbstractSurface

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLMultiSurfaceCoverage` *GMLMultiSurfaceCoverage*

Description

GMLMultiSurfaceCoverage

GMLMultiSurfaceCoverage

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML multisurface coverage

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractFeature` -> `geometa::GMLAbstractCoverage` -> `geometa::GMLAbstractDiscreteCoverage` -> `GMLMultiSurfaceCoverage`

Methods

Public methods:

- `GMLMultiSurfaceCoverage$new()`
- `GMLMultiSurfaceCoverage$clone()`

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

Usage:

```
GMLMultiSurfaceCoverage$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

```
GMLMultiSurfaceCoverage$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>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t
OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLObliqueCartesianCS` *GMLObliqueCartesianCS*

Description

GMLObliqueCartesianCS

GMLObliqueCartesianCS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLObliqueCartesianCS

Inherited Methods

`new(xml, defaults, id)` This method is used to instantiate a GML Abstract CRS

`addAxis(axis)` Adds an axis, object of class `GMLCoordinateSystemAxis`

`delAxis(axis)` Deletes an axis, object of class `GMLCoordinateSystemAxis`

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateSystem` -> `GMLObliqueCartesianCS`

Methods

Public methods:

- `GMLObliqueCartesianCS$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLOperationMethod` *GMLOperationMethod*

Description

GMLOperationMethod

GMLOperationMethod

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLOperationMethod

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `GMLOperationMethod`

Public fields

`formulaCitation` [`ISOCitation`]

`formula` [`GMLElement`]

`sourceDimensions` [`GMLElement`]

`targetDimensions` [`GMLElement`]

`parameter` [list of [`GMLOperationParameter` or `GMLOperationParameterGroup`]]

Methods

Public methods:

- `GMLOperationMethod$setFormulaCitation()`
- `GMLOperationMethod$setFormula()`
- `GMLOperationMethod$setSourceDimensions()`
- `GMLOperationMethod$setTargetDimensions()`
- `GMLOperationMethod$addParameter()`
- `GMLOperationMethod$delParameter()`
- `GMLOperationMethod$clone()`

Method `setFormulaCitation()`**:** Sets the formula citation

Usage:

`GMLOperationMethod$setFormulaCitation(citation)`

Arguments:

`citation` object of class `ISOCitation`

Method `setFormula()`**:** Set formula

Usage:

`GMLOperationMethod$setFormula(formula)`

Arguments:

`formula` formula, object of class character

Method `setSourceDimensions()`**:** Set source dimensions

Usage:

`GMLOperationMethod$setSourceDimensions(value)`

Arguments:

`value` value, object of class integer

Method `setTargetDimensions()`**:** Set target dimensions

Usage:

`GMLOperationMethod$setTargetDimensions(value)`

Arguments:

`value` value, object of class integer

Method `addParameter()`**:** Adds a parameter

Usage:

`GMLOperationMethod$addParameter(param)`

Arguments:

`param` object of class GMLOperationParameter or GMLOperationParameterGroup

Returns: `TRUE` if added, `FALSE` otherwise

Method `delParameter()`**:** Deletes a parameter

Usage:

`GMLOperationMethod$delParameter(param)`

Arguments:

`param` object of class GMLOperationParameter or GMLOperationParameterGroup

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`GMLOperationMethod$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLOperationParameter` *GMLOperationParameter*

Description

GMLOperationParameter

GMLOperationParameter

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLOperationParameter

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractGeneralOperationParameter` -> `GMLOperationParameter`

Methods

Public methods:

- `GMLOperationParameter$clone()`

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

Usage:

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

Arguments:

deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLOperationParameterGroup` *GMLOperationParameterGroup*

Description

GMLOperationParameterGroup

GMLOperationParameterGroup

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLOperationParameterGroup

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractGeneralOperationParameter` -> `GMLOperationParameterGroup`

Public fields

`maximumOccurs` maximumOccurs [0..1]: integer

`parameter` parameter [2..*]: GMLOperationParameter / GMLOperationParameterGroup

Methods

Public methods:

- `GMLOperationParameterGroup$setMaximumOccurs()`
- `GMLOperationParameterGroup$addParameter()`
- `GMLOperationParameterGroup$delParameter()`
- `GMLOperationParameterGroup$clone()`

Method `setMaximumOccurs()`**:** Set maximum occurs

Usage:

```
GMLOperationParameterGroup$setMaximumOccurs(maximumOccurs)
```

Arguments:

`maximumOccurs` maximumOccurs, object of class integer

Method `addParameter():` Adds a parameter

Usage:

```
GMLOperationParameterGroup$addParameter(param)
```

Arguments:

`param` object of class GMLOperationParameter or GMLOperationParameterGroup

Returns: `TRUE` if added, `FALSE` otherwise

Method `delParameter():` Deletes a parameter

Usage:

```
GMLOperationParameterGroup$delParameter(param)
```

Arguments:

`param` object of class GMLOperationParameter or GMLOperationParameterGroup

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLParameterValue` *GMLParameterValue*

Description

GMLParameterValue

GMLParameterValue

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML parameter value

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGeneralParameterValue` -> `GMLParameterValue`

Public fields

`value` value

`stringValue` string value

`integerValue` integer value

`booleanValue` boolean value

`valueList` value list

`integerValueList` integer value list

`valueFile` value file

`operationParameter` operation parameter

Methods

Public methods:

- `GMLParameterValue$new()`
- `GMLParameterValue$setValue()`
- `GMLParameterValue$setStringValue()`
- `GMLParameterValue$setIntegerValue()`
- `GMLParameterValue$setBooleanValue()`
- `GMLParameterValue$setValueFile()`
- `GMLParameterValue$setOperationParameter()`
- `GMLParameterValue$clone()`

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

Usage:

```
GMLParameterValue$new(xml = NULL, defaults = list())
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` default values

Method `setValue()`**:** Set value

Usage:

```
GMLParameterValue$setValue(value, uom)
```

Arguments:

`value` value, object of class numeric

`uom` uom

Method `setStringValue()`**:** Set string value

Usage:

```
GMLParameterValue$setStringValue(value)
```

Arguments:

`value` value

Method `setIntegerValue():` Set integer value

Usage:

```
GMLParameterValue$setIntegerValue(value)
```

Arguments:

`value` value, object of class integer

Method `setBooleanValue():` Set boolean value

Usage:

```
GMLParameterValue$setBooleanValue(value)
```

Arguments:

`value` object of class logical

Method `setValueFile():` Set value file

Usage:

```
GMLParameterValue$setValueFile(value)
```

Arguments:

`value` value

Method `setOperationParameter():` Set operation parameter

Usage:

```
GMLParameterValue$setOperationParameter(operationParameter)
```

Arguments:

`operationParameter` object of class GMLOperationParameter

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

Examples

```
gml <- GMLParameterValue$new()
gml$setValue(1.1, "test")
op <- GMLOperationParameter$new()
op$setDescriptionReference("someref")
op$setIdentifier("identifier", "codespace")
op$addName("name1", "codespace")
op$addName("name2", "codespace")
op$setMinimumOccurs(2L)
gml$setOperationParameter(op)
xml <- gml$encode()
```

GMLParameterValueGroup

GMLParameterValueGroup

Description

GMLParameterValueGroup

GMLParameterValueGroup

Format

R6Class object.

Value

Object of R6Class for modelling an GML parameter value group

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::GMLAbstractObject -> geometa::GMLAbstractGeneralParameterValue -> GMLParameterValueGroup

Public fields

parameterValue	parameter value list
group	group

Methods

Public methods:

- GMLParameterValueGroup$new()
- GMLParameterValueGroup$addParameterValue()
- GMLParameterValueGroup$delParameterValue()
- GMLParameterValueGroup$setOperationParameterGroup()

- `GMLParameterValueGroup$clone()`

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

Usage:

```
GMLParameterValueGroup$new(xml = NULL, defaults = list())
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` default values

Method `addParameterValue()`**:** Adds parameter value

Usage:

```
GMLParameterValueGroup$addParameterValue(parameterValue)
```

Arguments:

`parameterValue` parameter value, object of class GMLParameterValue

Returns: `TRUE` if added, `FALSE` otherwise

Method `delParameterValue()`**:** Deletes parameter value

Usage:

```
GMLParameterValueGroup$delParameterValue(parameterValue)
```

Arguments:

`parameterValue` parameter value, object of class GMLParameterValue

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setOperationParameterGroup()`**:** Set operation parameter group

Usage:

```
GMLParameterValueGroup$setOperationParameterGroup(operationParameterGroup)
```

Arguments:

`operationParameterGroup` operation parameter group

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

Examples

```
gml <- GMLParameterValueGroup$new()
```

GMLPoint — *GMLPoint*

Description

GMLPoint

GMLPoint

Format

R6Class object.

Value

Object of R6Class for modelling an GML point

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::GMLAbstractObject -> geometa::GMLAbstractGML -> geometa::GMLAbstractGeometry -> geometa::GMLAbstractGeometricPrimitive -> GMLPoint

Public fields

pos matrix of positions

Methods

Public methods:

- GMLPoint$new()
- GMLPoint$clone()

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

Usage:

```
GMLPoint$new(xml = NULL, sfg = NULL, m = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`sfg` simple feature geometry from **sf**

`m` simple object of class matrix

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLPolarCS` *GMLPolarCS*

Description

GMLPolarCS

GMLPolarCS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLPolarCS

Inherited Methods

`new(xml, defaults, id)` This method is used to instantiate a GML Abstract CRS

`addAxis(axis)` Adds an axis, object of class `GMLCoordinateSystemAxis`

`delAxis(axis)` Deletes an axis, object of class `GMLCoordinateSystemAxis`

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateSystem` -> `GMLPolarCS`

Methods

Public methods:

- `GMLPolarCS$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLPolygon` *GMLPoint*

Description

GMLPoint

GMLPoint

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML point

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractGeometricPrimitive` -> `geometa::GMLAbstractSurface` -> `GMLPolygon`

Public fields

`exterior` list of exterior polygons

`interior` list of interior polygons

Methods

Public methods:

- `GMLPolygon$new()`
- `GMLPolygon$clone()`

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

Usage:

```
GMLPolygon$new(xml = NULL, sfg)
```

Arguments:

`xml` object of class XMLInternalNode-class

`sfg` simple object from **sf**

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLProjectedCRS` *GMLProjectedCRS*

Description

GMLProjectedCRS

GMLProjectedCRS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLProjectedCRS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCRS` -> `geometa::GMLAbstractSingleCRS` -> `geometa::GMLAbstractGeneralDerivedCRS` -> `GMLProjectedCRS`

Public fields

`baseGeodeticCRS` baseGeodeticCRS [1..1]: GMLGeodeticCRS

`cartesianCS` cartesianCS [1..1]: GMLCartesianCS

Methods

Public methods:

- `GMLProjectedCRS$setBaseGeodeticCRS()`
- `GMLProjectedCRS$setCartesianCS()`
- `GMLProjectedCRS$clone()`

Method `setBaseGeodeticCRS()`**:** Set base Geodetic CRS

Usage:

```
GMLProjectedCRS$setBaseGeodeticCRS(crs)
```

Arguments:

`crs` crs, object of class GMLGeodeticCRS

Method `setCartesianCS()`**:** Set cartesian CRS. Not yet supported

Usage:

```
GMLProjectedCRS$setCartesianCS(cs)
```

Arguments:

`cs` cs, object of class `GMLCartesianCRS`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

GMLRectifiedGrid *GMLRectifiedGrid*

Description

GMLRectifiedGrid

GMLRectifiedGrid

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML rectified grid

Methods

`new(xml, element)` This method is used to instantiate a GML rectified grid

`setOrigin(x,y)` Set the origin of the rectified grid

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractImplicitGeometry` -> `geometa::GMLGrid` -> `GMLRectifiedGrid`

Public fields

`origin` origin

`offsetVector` offset vector

Methods

Public methods:

- `GMLRectifiedGrid$new()`
- `GMLRectifiedGrid$setOrigin()`
- `GMLRectifiedGrid$addOffsetVector()`
- `GMLRectifiedGrid$delOffsetVector()`
- `GMLRectifiedGrid$clone()`

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

Usage:

```
GMLRectifiedGrid$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setOrigin()`**:** Set origin

Usage:

```
GMLRectifiedGrid$setOrigin(x, y)
```

Arguments:

`x` x

`y` y

Method `addOffsetVector()`**:** Adds offset vector

Usage:

```
GMLRectifiedGrid$addOffsetVector(vec)
```

Arguments:

`vec` vec, object of class vector

Returns: `TRUE` if added, `FALSE` otherwise

Method `delOffsetVector()`**:** Deletes offset vector

Usage:

```
GMLRectifiedGrid$delOffsetVector(vec)
```

Arguments:

`vec` vec, object of class vector

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

GMLRectifiedGridCoverage *GMLRectifiedGridCoverage*

Description

GMLRectifiedGridCoverage

GMLRectifiedGridCoverage

Format

R6Class object.

Value

Object of R6Class for modelling an GML rectified grid coverage

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::GMLAbstractObject -> geometa::GMLAbstractFeature -> geometa::GMLAbstractCoverage -> geometa::GMLAbstractDiscreteCoverage -> GMLRectifiedGridCoverage

Methods

Public methods:

- GMLRectifiedGridCoverage$new()
- GMLRectifiedGridCoverage$clone()

Method new()**:** Initializes object

Usage:

```
GMLRectifiedGridCoverage$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t
OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLReferenceableGridByArray` *GMLReferenceableGridByArray*

Description

GMLReferenceableGridByArray

GMLReferenceableGridByArray

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML ReferenceableGridByArray

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractImplicitGeometry` -> `geometa::GMLGrid` -> `geometa::GMLAbstractReferenceableGrid` -> `GMLReferenceableGridByArray`

Public fields

`generalGridAxis` general grid axis

Methods

Public methods:

- `GMLReferenceableGridByArray$new()`
- `GMLReferenceableGridByArray$clone()`

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

Usage:

```
GMLReferenceableGridByArray$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

```
GMLReferenceableGridByArray$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>

References

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

OGC GML 3.3 Schema. http://schemas.opengis.net/gml/3.3/referenceableGrid.xsd

GMLReferenceableGridByTransformation
GMLReferenceableGridByTransformation

Description

GMLReferenceableGridByTransformation

GMLReferenceableGridByTransformation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML ReferenceableGridByTransformation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractImplicitGeometry` -> `geometa::GMLGrid` -> `geometa::GMLAbstractReferenceableGrid` -> `GMLReferenceableGridByTransformation`

Public fields

`transformation` transformation

`concatenatedOperation` concatenated operation

Methods

Public methods:

- `GMLReferenceableGridByTransformation$new()`
- `GMLReferenceableGridByTransformation$clone()`

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

Usage:

```
GMLReferenceableGridByTransformation$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

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

Usage:

```
GMLReferenceableGridByTransformation$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>

References

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

OGC GML 3.3 Schema. http://schemas.opengis.net/gml/3.3/referenceableGrid.xsd

---

`GMLReferenceableGridByVectors` *GMLReferenceableGridByVectors*

Description

GMLReferenceableGridByVectors

GMLReferenceableGridByVectors

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML ReferenceableGridByVectors

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractGeometry` -> `geometa::GMLAbstractImplicitGeometry` -> `geometa::GMLGrid` -> `geometa::GMLAbstractReferenceableGrid` -> `GMLReferenceableGridByVectors`

Public fields

`origin` origin

`generalGridAxis` general grid axis

Methods

Public methods:

- `GMLReferenceableGridByVectors$new()`
- `GMLReferenceableGridByVectors$setOrigin()`
- `GMLReferenceableGridByVectors$addGeneralGridAxis()`
- `GMLReferenceableGridByVectors$delGeneralGridAxis()`
- `GMLReferenceableGridByVectors$clone()`

Method `new()`: Initializes object

Usage:

```
GMLReferenceableGridByVectors$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`attrs` list of attributes

`defaults` list of default values

`wrap` wrap element?

Method `setOrigin()`: Set origin

Usage:

```
GMLReferenceableGridByVectors$setOrigin(coords)
```

Arguments:

`coords` coords, object of class list

Method `addGeneralGridAxis()`: Adds general grid axis

Usage:

```
GMLReferenceableGridByVectors$addGeneralGridAxis(axis)
```

Arguments:

`axis` object of class GMLGeneralGridAxis

Returns: `TRUE` if added, `FALSE` otherwise

Method `delGeneralGridAxis()`: Deletes general grid axis

Usage:

```
GMLReferenceableGridByVectors$delGeneralGridAxis(axis)
```

Arguments:

`axis` object of class GMLGeneralGridAxis

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

```
GMLReferenceableGridByVectors$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>

References

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

OGC GML 3.3 Schema. http://schemas.opengis.net/gml/3.3/referenceableGrid.xsd

---

`GMLSphericalCS` *GMLSphericalCS*

Description

GMLSphericalCS

GMLSphericalCS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLSphericalCS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateSystem` -> `GMLSphericalCS`

Methods

Public methods:

- `GMLSphericalCS$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLTemporalCRS` *GMLTemporalCRS*

Description

GMLTemporalCRS

GMLTemporalCRS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLTemporalCRS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCRS` -> `geometa::GMLAbstractSingleCRS` -> `GMLTemporalCRS`

Public fields

`timeCS` time CS

`temporalDatum` temporal datum

Methods

Public methods:

- `GMLTemporalCRS$setTimeCS()`
- `GMLTemporalCRS$setTemporalDatum()`
- `GMLTemporalCRS$clone()`

Method `setTimeCS()`: Set time CS

Usage:

```
GMLTemporalCRS$setTimeCS(timeCS)
```

Arguments:

`timeCS` time CS, object of class GMLTimeCS

Method `setTemporalDatum()`: Set temporal datum

Usage:

```
GMLTemporalCRS$setTemporalDatum(temporalDatum)
```

Arguments:

`temporalDatum` temporal datum

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLTemporalCS` *GMLTemporalCS*

Description

GMLTemporalCS

GMLTemporalCS

Format

`R6Class` object.

Value

Object of R6Class for modelling an GMLTemporalCS

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::GMLAbstractObject -> geometa::GMLAbstractGML -> geometa::GMLDefinition -> geometa::GMLAbstractCoordinateSystem -> GMLTemporalCS

Methods

Public methods:

- GMLTemporalCS$clone()

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

GMLTimeCS *GMLTimeCS*

Description

GMLTimeCS

GMLTimeCS

Format

R6Class object.

Value

Object of R6Class for modelling an GMLTimeCS

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::GMLAbstractObject -> geometa::GMLAbstractGML -> geometa::GMLDefinition -> geometa::GMLAbstractCoordinateSystem -> GMLTimeCS

Methods

Public methods:

- GMLTimeCS$clone()

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLTimeInstant` *GMLTimeInstant*

Description

GMLTimeInstant

GMLTimeInstant

Format

R6Class object.

Value

Object of R6Class for modelling an GMLTimeInstant

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::GMLAbstractObject -> geometa::GMLAbstractGML -> geometa::GMLAbstractTimeObject -> geometa::GMLAbstractTimePrimitive -> geometa::GMLAbstractTimeGeometricPrimitive -> GMLTimeInstant

Public fields

`timePosition` [numeric|character|Date|POSIXt]

Methods

Public methods:

- `GMLTimeInstant$new()`
- `GMLTimeInstant$setTimePosition()`
- `GMLTimeInstant$toISOFormat()`
- `GMLTimeInstant$clone()`

Method `new()`: Initializes object

Usage:

```
GMLTimeInstant$new(xml = NULL, timePosition)
```

Arguments:

`xml` object of class XMLInternalNode-class

`timePosition` time position

Method `setTimePosition()`: Sets the position (date or date and time of the resource contents),

Usage:

```
GMLTimeInstant$setTimePosition(timePosition)
```

Arguments:

`timePosition` object of class "numeric", "POSIXct"/"POSIXt" or "Date"

Method `toISOFormat()`: Export to ISO format (character)

Usage:

```
GMLTimeInstant$toISOFormat()
```

Returns: a character in ISO format

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```
time <- ISOdate(2000, 1, 12, 12, 59, 45)
md <- GMLTimeInstant$new(timePosition = time)
xml <- md$encode()
```

GMLTimePeriod *GMLTimePeriod*

Description

GMLTimePeriod

GMLTimePeriod

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLTimePeriod

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLAbstractTimeObject` -> `geometa::GMLAbstractTimePrimitive` -> `geometa::GMLAbstractTimeGeometricPrimitive` -> `GMLTimePeriod`

Public fields

`beginPosition` beginPosition [1]: 'POSIXct','POSIXt'

`endPosition` endPosition [1]: 'POSIXct','POSIXt'

`duration` duration [0..1]: character

Methods

Public methods:

- `GMLTimePeriod$new()`
- `GMLTimePeriod$setBeginPosition()`
- `GMLTimePeriod$setEndPosition()`
- `GMLTimePeriod$computeInterval()`
- `GMLTimePeriod$setDuration()`
- `GMLTimePeriod$clone()`

Method `new()`: Initializes object

Usage:

```
GMLTimePeriod$new(xml = NULL, beginPosition = NULL, endPosition = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`beginPosition` object of class numeric, Date or POSIXct-class `endPosition` object of class numeric, Date or POSIXct-class

Method `setBeginPosition()`**:** Set begin position

Usage:

```
GMLTimePeriod$setBeginPosition(beginPosition)
```

Arguments:

`beginPosition` object of class numeric, Date or POSIXct-class

Method `setEndPosition()`**:** Set end position

Usage:

```
GMLTimePeriod$setEndPosition(endPosition)
```

Arguments:

`endPosition` object of class numeric, Date or POSIXct-class

Method `computeInterval()`**:** Compute interval (ISO defined duration) and set proper attribute for XML encoding. The method calls the static function `GMLTimePeriod$computeISODuration`

Usage:

```
GMLTimePeriod$computeInterval()
```

Method `setDuration()`**:** Set ISO duration

Usage:

```
GMLTimePeriod$setDuration(
  years = 0,
  months = 0,
  days = 0,
  hours = 0,
  mins = 0,
  secs = 0
)
```

Arguments:

`years` years

`months` months

`days` days

`hours` hours

`mins` mins

`secs` secs

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```
start <- ISOdate(2000, 1, 12, 12, 59, 45)
end <- ISOdate(2010, 8, 22, 13, 12, 43)
md <- GMLTimePeriod$new(beginPosition = start, endPosition = end)
xml <- md$encode()
```

`GMLUnitDefinition` *GMLUnitDefinition*

Description

GMLUnitDefinition

GMLUnitDefinition

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML unit definition

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `GMLUnitDefinition`

Public fields

`quantityTypeReference` quantityTypeReference [0..1]: character

`catalogSymbol` catalogSymbol [0..1]: character

Methods

Public methods:

- `GMLUnitDefinition$new()`
- `GMLUnitDefinition$setQuantityTypeReference()`
- `GMLUnitDefinition$setCatalogSymbol()`
- `GMLUnitDefinition$clone()`

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

Usage:

`GMLUnitDefinition$new(xml = NULL, defaults = list(), id = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of default values

`id` id

Method `setQuantityTypeReference()`**:** Set quantity type reference. Content is reference to a remote value

Usage:

`GMLUnitDefinition$setQuantityTypeReference(quantityTypeReference)`

Arguments:

`quantityTypeReference` quantity type reference

Method `setCatalogSymbol()`**:** Set catalog symbol

Usage:

`GMLUnitDefinition$setCatalogSymbol(catalogSymbol)`

Arguments:

`catalogSymbol` catalog symbol, preferred lexical symbol used for this unit of measure

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

Usage:

`GMLUnitDefinition$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

Examples

```
gml <- GMLUnitDefinition$new()
gml$setDescriptionReference("someref")
gml$setIdentifier("identifier", "codespace")
gml$addName("name1", "codespace")
gml$addName("name2", "codespace")
gml$setQuantityTypeReference("someref")
gml$setCatalogSymbol("symbol")
```

`GMLUserDefinedCS` *GMLUserDefinedCS*

Description

GMLUserDefinedCS

GMLUserDefinedCS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLUserDefinedCS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateSystem` -> `GMLUserDefinedCS`

Methods

Public methods:

- `GMLUserDefinedCS$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

`GMLVerticalCRS` *GMLVerticalCRS*

Description

GMLVerticalCRS

GMLVerticalCRS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLVerticalCRS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCRS` -> `geometa::GMLAbstractSingleCRS` -> `GMLVerticalCRS`

Public fields

`verticalCS` [`GMLVerticalCS`]

`verticalDatum` [codeGMLVerticalDatum]

Methods

Public methods:

- `GMLVerticalCRS$setVerticalCS()`
- `GMLVerticalCRS$setVerticalDatum()`
- `GMLVerticalCRS$clone()`

Method `setVerticalCS()`**:** Set vertical CS

Usage:

```
GMLVerticalCRS$setVerticalCS(verticalCS)
```

Arguments:

`verticalCS` object of class GMLVerticalCS

Method `setVerticalDatum()`**:** Set vertical datum. not yet supported

Usage:

```
GMLVerticalCRS$setVerticalDatum(verticalDatum)
```

Arguments:

`verticalDatum` object of class `GMLVerticalDatum`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`GMLVerticalCS` *GMLVerticalCS*

Description

GMLVerticalCS

GMLVerticalCS

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GMLVerticalCS

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `geometa::GMLAbstractGML` -> `geometa::GMLDefinition` -> `geometa::GMLAbstractCoordinateSystem` -> `GMLVerticalCS`

Methods

Public methods:

- `GMLVerticalCS$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19136:2007 Geographic Information – Geographic Markup Language. http://www.iso.org/iso/iso_catalogue/catalogue_t

OGC Geography Markup Language. http://www.opengeospatial.org/standards/gml

---

`INSPIREMetadataValidator` *INSPIREMetadataValidator*

Description

INSPIREMetadataValidator

INSPIREMetadataValidator

Format

`R6Class` object.

Value

Object of `R6Class` for setting an INSPIREMetadataValidator

Super class

`geometa::geometaLogger` -> `INSPIREMetadataValidator`

Public fields

`url` url of the INSPIRE metadata validator

`running` wether the service is up and running

Methods

Public methods:

- `INSPIREMetadataValidator$new()`
- `INSPIREMetadataValidator$uploadFile()`
- `INSPIREMetadataValidator$getAPIKey()`
- `INSPIREMetadataValidator$getValidationReport()`
- `INSPIREMetadataValidator$clone()`

Method `new()`**:** Method used to instantiate an INSPIRE Metadata validator. To check metadata with the INSPIRE metadata validator, a user API key is now required, and should be specified with the `apiKey`. By default, the `url` will be the INSPIRE production service `https://inspire.ec.europa.eu/validator/swagger-ui.html`.

The `keyring_backend` can be set to use a different backend for storing the INSPIRE metadata validator API key with **keyring** (Default value is 'env').

Usage:

```
INSPIREMetadataValidator$new(
  url = "https://inspire.ec.europa.eu/validator/v2",
  apiKey,
  keyring_backend = "env"
)
```

Arguments:

`url` url

`apiKey` API key

`keyring_backend` backend name to use with **keyring** to store API key

Method `uploadFile()`**:** Uploads a file. Upload a XML metadata file to INSPIRE web-service. Method called internally through `getValidationReport`.

Usage:

```
INSPIREMetadataValidator$uploadFile(path)
```

Arguments:

`path` path

Returns: the response from the web-service

Method `getAPIKey()`**:** Retrieves the API key

Usage:

```
INSPIREMetadataValidator$getAPIKey()
```

Returns: the API key as character

Method `getValidationReport()`**:** Get validation report for a metadata specified either as R object of class ISOMetadata (from **geometa** package) or XMLInternalNode-class (from **XML** package), or as XML file, providing the path of the XML file to be sent to the INSPIRE metadata validator web-service. By default, a summary report is returned. To append the raw response of INSPIRE validation web-service to the summary report, set `raw = TRUE`.

Usage:

```
INSPIREMetadataValidator$getValidationReport(
  obj = NULL,
  file = NULL,
  raw = FALSE
)
```

Arguments:

`obj` obj

`file` file

`raw` raw

Returns: an object of class list

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

Usage:

`INSPIREMetadataValidator$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

INSPIRE Reference Validator Web Service (https://inspire.ec.europa.eu/validator/swagger-ui.html)

Examples

```
apiKey <- ""
if(nzchar(apiKey)){
  inspireValidator <- INSPIREMetadataValidator$new(apiKey = apiKey)
  inspireReport <- inspireValidator$getValidationReport(obj = ISOMetadata$new())
}
```

`ISOAbsoluteExternalPositionalAccuracy`
ISOAbsoluteExternalPositionalAccuracy

Description

ISOAbsoluteExternalPositionalAccuracy

ISOAbsoluteExternalPositionalAccuracy

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAbsoluteExternalPositionalAccuracy

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `geometa::ISOAbstractPositionalAccuracy` -> `ISOAbsoluteExternalPositionalAccuracy`

Methods

Public methods:

- `ISOAbsoluteExternalPositionalAccuracy$clone()`

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

Usage:

`ISOAbsoluteExternalPositionalAccuracy$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOAbsoluteExternalPositionalAccuracy$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

`ISOAbstractAggregate` *ISOAbstractAggregate*

Description

ISOAbstractAggregate

ISOAbstractAggregate

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAbstractAggregate

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOAbstractAggregate`

Public fields

`composedOf`	composedOf [1..*]
`seriesMetadata`	seriesMetadata [1..*]
`subset`	subset [0..*]
`superset`	superset [0..*]

Methods

Public methods:

- `ISOAbstractAggregate$new()`
- `ISOAbstractAggregate$addComposedOf()`
- `ISOAbstractAggregate$delComposedOf()`
- `ISOAbstractAggregate$addSeriesMetadata()`
- `ISOAbstractAggregate$delSeriesMetadata()`
- `ISOAbstractAggregate$addSubset()`
- `ISOAbstractAggregate$delSubset()`
- `ISOAbstractAggregate$addSuperset()`
- `ISOAbstractAggregate$delSuperset()`
- `ISOAbstractAggregate$clone()`

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

Usage:

```
ISOAbstractAggregate$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addComposedOf()`**:** Adds a dataset 'composedOf' relationship

Usage:

```
ISOAbstractAggregate$addComposedOf(composedOf)
```

Arguments:

`composedOf` object of class ISODataSet

Returns: `TRUE` if added, `FALSE` otherwise

Method `delComposedOf()`**:** Deletes a dataset 'composedOf' relationship

Usage:

```
ISOAbstractAggregate$delComposedOf(composedOf)
```

Arguments:

`composedOf` object of class ISODataSet

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addSeriesMetadata()`**:** Adds a series metadata

Usage:

```
ISOAbstractAggregate$addSeriesMetadata(metadata)
```

Arguments:

`metadata` object of class ISOMetadata

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSeriesMetadata()`**:** Deletes a series metadata

Usage:

```
ISOAbstractAggregate$delSeriesMetadata(metadata)
```

Arguments:

`metadata` object of class ISOMetadata

Returns: `TRUE` if added, `FALSE` otherwise

Method `addSubset()`**:** Adds subset

Usage:

```
ISOAbstractAggregate$addSubset(subset)
```

Arguments:

`subset` object of class inheriting ISOAbstractAggregate

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSubset()`**:** Deletes subset

Usage:

```
ISOAbstractAggregate$delSubset(subset)
```

Arguments:

`subset` object of class inheriting ISOAbstractAggregate

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addSuperset():` Adds superset

Usage:

```
ISOAbstractAggregate$addSuperset(superset)
```

Arguments:

`superset` object of class inheriting ISOAbstractAggregate

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSuperset():` Deletes superset

Usage:

```
ISOAbstractAggregate$delSuperset(superset)
```

Arguments:

`superset` object of class inheriting ISOAbstractAggregate

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

abstract class

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOAbstractCarrierOfCharacteristics` *ISOAbstractCarrierOfCharacteristics*

Description

ISOAbstractCarrierOfCharacteristics

ISOAbstractCarrierOfCharacteristics

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an abstract ISOCarrierOfCharacteristics

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOAbstractCarrierOfCharacteristics`

Public fields

`featureType` featureType [0..1]: ISOFeatureType

`constrainedBy` constrainedBy [0..*]: ISOConstraint

Methods

Public methods:

- `ISOAbstractCarrierOfCharacteristics$new()`
- `ISOAbstractCarrierOfCharacteristics$setFeatureType()`
- `ISOAbstractCarrierOfCharacteristics$addConstraint()`
- `ISOAbstractCarrierOfCharacteristics$delConstraint()`
- `ISOAbstractCarrierOfCharacteristics$clone()`

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

Usage:

```
ISOAbstractCarrierOfCharacteristics$new(xml = NULL, defaults = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` default values

Method `setFeatureType()`**:** Set feature type

Usage:

```
ISOAbstractCarrierOfCharacteristics$setFeatureType(featureType)
```

Arguments:

`featureType` feature type, object of class ISOFeatureType

Method `addConstraint()`**:** Adds constraint

Usage:

```
ISOAbstractCarrierOfCharacteristics$addConstraint(constraint)
```

Arguments:

`constraint,` object of class ISOConstraint

Returns: `TRUE` if added, FALSE otherwise

Method `delConstraint()`**:** Deletes constraint

Usage:

```
ISOAbstractCarrierOfCharacteristics$delConstraint(constraint)
```

Arguments:

`constraint,` object of class ISOConstraint

Returns: `TRUE` if deleted, FALSE otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

abstract class

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

`ISOAbstractCatalogue` *ISOAbstractCatalogue*

Description

ISOAbstractCatalogue

ISOAbstractCatalogue

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAbstracCatalogue

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOAbstractCatalogue`

Public fields

`name` name [1..1]: character

`scope` scope [1..*]: character

`fieldOfApplication` fieldOfApplication [0.*]: character

`versionNumber` versionNumber [1..1]: character

`versionDate` versionDate [1..1]: Date/Posix

Methods

Public methods:

- `ISOAbstractCatalogue$new()`
- `ISOAbstractCatalogue$setName()`
- `ISOAbstractCatalogue$addScope()`
- `ISOAbstractCatalogue$delScope()`
- `ISOAbstractCatalogue$addFieldOfApplication()`
- `ISOAbstractCatalogue$delFieldOfApplication()`
- `ISOAbstractCatalogue$setVersionNumber()`
- `ISOAbstractCatalogue$setVersionDate()`
- `ISOAbstractCatalogue$clone()`

Method `new()`: Initializes object

Usage:

```
ISOAbstractCatalogue$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setName():` Sets the name. Locale names can be specified as

Usage:

```
ISOAbstractCatalogue$setName(name, locales = NULL)
```

Arguments:

`name` name

`locales` locales, object of class list

Method `addScope():` Adds scope

Usage:

```
ISOAbstractCatalogue$addScope(scope, locales = NULL)
```

Arguments:

`scope` scope

`locales` locales, object of class list

Returns: `TRUE` if added, `FALSE` otherwise

Method `delScope():` Deletes scope

Usage:

```
ISOAbstractCatalogue$delScope(scope, locales = NULL)
```

Arguments:

`scope` scope

`locales` locales, object of class list

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addFieldOfApplication():` Adds field of application

Usage:

```
ISOAbstractCatalogue$addFieldOfApplication(fieldOfApplication, locales = NULL)
```

Arguments:

`fieldOfApplication` field of application

`locales` locales, object of class list

Returns: `TRUE` if added, `FALSE` otherwise

Method `delFieldOfApplication():` Deletes field of application

Usage:

```
ISOAbstractCatalogue$delFieldOfApplication(fieldOfApplication)
```

Arguments:

`fieldOfApplication` field of application

`locales` locales, object of class list

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setVersionNumber():` Set version number

Usage:

```
ISOAbstractCatalogue$setVersionNumber(versionNumber)
```

Arguments:

`versionNumber` version number

Method `setVersionDate():` Set version date

Usage:

```
ISOAbstractCatalogue$setVersionDate(versionDate)
```

Arguments:

`versionDate` version date

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19139:2007 Metadata - XML schema implementation

`ISOAbstractCompleteness` *ISOAbstractCompleteness*

Description

ISOAbstractCompleteness

ISOAbstractCompleteness

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAbstractCompleteness

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `ISOAbstractCompleteness`

Methods

Public methods:

- `ISOAbstractCompleteness$clone()`

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

Usage:

`ISOAbstractCompleteness$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOAbstractGenericName` *ISOAbstractGenericName*

Description

ISOAbstractGenericName

ISOAbstractGenericName

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO abstract GenericName

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLCodeType` -> `ISOAbstractGenericName`

Public fields

`value` value

Methods

Public methods:

- `ISOAbstractGenericName$new()`
- `ISOAbstractGenericName$clone()`

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

Usage:

```
ISOAbstractGenericName$new(xml = NULL, value = NULL, codeSpace = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`codeSpace` code space

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOAbstractLogicalConsistency`

ISOAbstractLogicalConsistency

Description

ISOAbstractLogicalConsistency

ISOAbstractLogicalConsistency

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAbstractLogicalConsistency

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `ISOAbstractLogicalConsistency`

Methods

Public methods:

- `ISOAbstractLogicalConsistency$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOAbstractObject` *ISOAbstractObject*

Description

ISOAbstractObject

ISOAbstractObject

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Metadata Element

Static Methods

`getISOStandardByPrefix(prefix)` Inherit the ISO (and/or OGC) standard reference for a given standard prefix (e.g. GMD). The object returned is a `data.frame` containing the specification reference and title.

`getISOStandard(clazz)` Inherit the ISO (and/or OGC) standard reference for a given **geometa** class. The object returned is a `data.frame` containing the specification reference and title.

`getISOClasses(extended, pretty)` Get the list of classes supported by **geometa**. By default, `extended` is set to `FALSE` (restrained to **geometa** environment). If `TRUE`, this allows to list eventual classes loaded in your global environment and that extend **geometa** classes. The argument `pretty` gives a the list of classes and associated ISO/OGC standard information as `data.frame`.

`getISOClassByNode(node)` Inherit the ISO class matching an XML document or node

`compare(metadataElement1, metadataElement2)` Compares two metadata elements objects. Returns TRUE if they are equal, FALSE otherwise. The comparison of object is done by comparing the XML representation of the objects (since no R6 object comparison method seems to exist)

Abstract Methods

`new(xml, element, namespace, defaults, attrs)` This method is used to instantiate an ISOAbstractObject

`print()` Provides a custom print output (as tree) of the current class

`decode(xml)` Decodes a ISOMetadata* R6 object from XML representation

`encode(addNS, validate, strict, inspire, inspireValidator, resetSerialID, setSerialID, encoding)` Encodes a ISOMetadata* R6 object to XML representation. By default, namespace definition will be added to XML root (`addNS = TRUE`), and validation of object will be performed (`validate = TRUE`) prior to its XML encoding. The argument `strict` allows to stop the encoding in case object is not valid, with a default value set to `FALSE`. The argument `setSerialID` is used by **geometa** to generate automatically serial IDs associated to XML elements, in particular for GML, default value is `TRUE` (recommended value). The argument `resetSerialID` is used by **geometa** for reseting mandatory IDs associated to XML elements, such as GML objects, default value is `TRUE` (recommended value). Setting `inspire` to TRUE (default FALSE), the metadata will be checked with the INSPIRE metadata validator (online web-service provided by INSPIRE). To check metadata with the INSPIRE metadata validator, setting an INSPIRE metadata validator is now required, and should be specified with the `inspireValidator`. See `INSPIREMetadataValidator` for more details

`validate(xml, strict, inspire, inspireValidator)` Validates the encoded XML against ISO 19139 XML schemas. If `strict` is `TRUE`, a error will be raised. Default is `FALSE`. Setting `inspire` to`TRUE` (default `FALSE`), the metadata will be checked with the INSPIRE metadata validator (online web-service provided by INSPIRE). To check metadata with the INSPIRE metadata validator, setting an INSPIRE metadata validator is now required, and should be specified with the `inspireValidator`. See `INSPIREMetadataValidator` for more details

`save(file, ...)` Saves the current metadata object XML representation to a file. This utility ensures proper indentation of XML file produced. Additional parameters from `$encode()` method can be specified, such as `inspire` to check the INSPIRE metadata validity.

`getNamespaceDefinition(recursive)` Gets the namespace definition of the current ISO* class. By default, only the namespace definition of the current element is retrieved (`recursive = FALSE`).

`getClassName(level)` Gets the class name. The level of class inheritance. Default is `1L`

`getClass()` Gets the class

`wrapBaseElement(field, fieldObj)` Wraps a base element type

`setIsNull(isNull, reason)` Sets the object as null object for the XML. In case `isNull` is `TRUE`, a reason should be specified among values 'inapplicable', 'missing', 'template', 'unknown', 'withheld'. By default, the reason is set 'missing'.

`contains(field, metadataElement)` Indicates of the present class object contains an metadata element object for a particular list-based field.

`addListElement(field, metadataElement)` Adds a metadata element to a list-based field. Returns TRUE if the element has been added, FALSE otherwise. In case an element is already added, the element will not be added and this method will return FALSE.

`delListElement(field, metadataElement)` Deletes a metadata element from a list-based field. Returns TRUE if the element has been deleted, FALSE otherwise. In case an element is abstent, this method will return FALSE.

`setAttr(attrKey, attrValue)` Set an attribute

`addFieldAttrs(field, ...)` Allows to add one more xlink attributes a field (element property)

`setId(id, addNS)` Set an id. By default `addNS` is `FALSE` (no namespace prefix added).

`setHref(href)` Sets an href reference

`setCodeList(codeList)` Sets a codeList

`setCodeListValue(codeListValue)` Sets a codeList value

`setCodeSpace(codeSpace)` Set a codeSpace

`setValue(value)` Set a value

`isDocument()` Indicates if the object is a metadata document, typically an object of class `ISOMetadata` or `ISOFeatureCatalogue`

`isFieldInheritedFrom(field)` Gives the parent from which the field is inherited, otherwise return `NULL`.

`createLocalisedProperty(text, locales)` Creates a localised property made of a default text and a list of localised texts.

Super class

`geometa::geometaLogger` -> `ISOAbstractObject`

Public fields

`wrap` wrap

`element` element

`namespace` namespace

`defaults` defaults

`attrs` attributes

`printAttrs` attributes to print

`parentAttrs` parent attributes

`value` value

`value_as_field` value as field?

`isNull` is null?

`anyElement` any element?

Methods

Public methods:

- `ISOAbstractObject$new()`
- `ISOAbstractObject$print()`
- `ISOAbstractObject$decode()`
- `ISOAbstractObject$encode()`
- `ISOAbstractObject$validate()`
- `ISOAbstractObject$save()`
- `ISOAbstractObject$getNamespaceDefinition()`
- `ISOAbstractObject$getClassName()`
- `ISOAbstractObject$getClass()`
- `ISOAbstractObject$wrapBaseElement()`
- `ISOAbstractObject$setIsNull()`
- `ISOAbstractObject$contains()`
- `ISOAbstractObject$addListElement()`
- `ISOAbstractObject$delListElement()`
- `ISOAbstractObject$setAttr()`
- `ISOAbstractObject$addFieldAttrs()`
- `ISOAbstractObject$setId()`
- `ISOAbstractObject$setHref()`
- `ISOAbstractObject$setCodeList()`
- `ISOAbstractObject$setCodeListValue()`
- `ISOAbstractObject$setCodeSpace()`
- `ISOAbstractObject$setValue()`
- `ISOAbstractObject$isDocument()`
- `ISOAbstractObject$isFieldInheritedFrom()`
- `ISOAbstractObject$createLocalisedProperty()`

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

Usage:

```
ISOAbstractObject$new(
  xml = NULL,
  element = NULL,
  namespace = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE,
  value_as_field = FALSE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`element` element name

`namespace` namespace

`attrs` attrs

`defaults` defaults

`wrap` wrap?

`value_as_field` value as field?

Method `print()`**:** Provides a custom print output (as tree) of the current class

Usage:

```
ISOAbstractObject$print(..., depth = 1)
```

Arguments:

`...` args

`depth` class nesting depth

Method `decode()`**:** Decodes object from XML

Usage:

```
ISOAbstractObject$decode(xml)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `encode()`**:** Encodes object as XML.

By default, namespace definition will be added to XML root (`addNS = TRUE`), and validation of object will be performed (`validate = TRUE`) prior to its XML encoding. The argument `strict` allows to stop the encoding in case object is not valid, with a default value set to `FALSE`.

The argument `setSerialID` is used by **geometa** to generate automatically serial IDs associated to XML elements, in particular for GML, default value is `TRUE` (recommended value).

The argument `resetSerialID` is used by **geometa** for reseting mandatory IDs associated to XML elements, such as GML objects, default value is `TRUE` (recommended value).

Setting `inspire` to TRUE (default FALSE), the metadata will be checked with the INSPIRE metadata validator (online web-service provided by INSPIRE). To check metadata with the INSPIRE metadata validator, setting an INSPIRE metadata validator is now required, and should be specified with the `inspireValidator`. See `INSPIREMetadataValidator` for more details

Usage:

```
ISOAbstractObject$encode(
  addNS = TRUE,
  validate = TRUE,
  strict = FALSE,
  inspire = FALSE,
  inspireValidator = NULL,
  resetSerialID = TRUE,
  setSerialID = TRUE,
  encoding = "UTF-8"
)
```

Arguments:

`addNS` add namespace? Default is `TRUE`

`validate` validate XML output against schemas?

`strict` strict validation? Default is `FALSE`.

`inspire` perform INSPIRE validation? Default is `FALSE`

`inspireValidator` an object of class INSPIREMetadataValidator to perform INSPIRE metadata validation

`resetSerialID` reset Serial ID? Default is `TRUE`

`setSerialID` set serial ID? Default is `TRUE`

`encoding` encoding. Default is `UTF-8`

Method `validate()`**:** Validates an XML object resulting from object encoding

Usage:

```
ISOAbstractObject$validate(
  xml = NULL,
  strict = FALSE,
  inspire = FALSE,
  inspireValidator = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`strict` strict validation? If `TRUE`, a invalid XML will return an error

`inspire` perform INSPIRE validation? Default is `FALSE`

`inspireValidator` an object of class INSPIREMetadataValidator to perform INSPIRE metadata validation

Returns: `TRUE` if valid, `FALSE` otherwise

Method `save()`**:** Save XML representation resulting from `$encode(...)` method to a file

Usage:

```
ISOAbstractObject$save(file, ...)
```

Arguments:

`file` file

`...` any other argument from `$encode(...)` method

Method `getNamespaceDefinition()`**:** Get namespace definition

Usage:

```
ISOAbstractObject$getNamespaceDefinition(recursive = FALSE)
```

Arguments:

`recursive` recursive namespace definitions? Default is `FALSE`

Returns: the list of XML namespace definitions

Method `getClassName():` Get class name

Usage:

```
ISOAbstractObject$getClassName(level = 1L)
```

Arguments:

`level` level of class

Returns: the class name

Method `getClass():` Get class

Usage:

```
ISOAbstractObject$getClass()
```

Returns: the corresponding class, as R6Class reference object generator

Method `wrapBaseElement():` Wraps base element

Usage:

```
ISOAbstractObject$wrapBaseElement(field, fieldObj)
```

Arguments:

`field` field name

`fieldObj` field object

`an` object of class R6Class

Method `setIsNull():` Set Is Null

Usage:

```
ISOAbstractObject$setIsNull(isNull, reason = "missing")
```

Arguments:

`isNull` object of class logical

`reason` reason why object is Null

Method `contains():` Util to know if a field contain a metadata element

Usage:

```
ISOAbstractObject$contains(field, metadataElement)
```

Arguments:

`field` field name

`metadataElement` metadata element

Returns: `TRUE` if contains, `FALSE` otherwise

Method `addListElement():` Util to add an element to a list of elements for N cardinality of a target element name

Usage:

`ISOAbstractObject$addListElement(field, metadataElement)`

Arguments:

`field` field

`metadataElement` metadata element

Returns: `TRUE` if added, `FALSE` otherwise

Method `delListElement()`**:** Util to deleted an element to a list of elements for N cardinality of a target element name

Usage:

`ISOAbstractObject$delListElement(field, metadataElement)`

Arguments:

`field` field

`metadataElement` metadata element

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setAttr()`**:** Util to set an attribute

Usage:

`ISOAbstractObject$setAttr(attrKey, attrValue)`

Arguments:

`attrKey` attribute key

`attrValue` attribute value

Method `addFieldAttrs()`**:** Util add field attributes, over the XML field wrapping element instead of the element itself

Usage:

`ISOAbstractObject$addFieldAttrs(field, ...)`

Arguments:

`field` field

`...` list of attributes

Method `setId()`**:** Set id

Usage:

`ISOAbstractObject$setId(id, addNS = FALSE)`

Arguments:

`id` id

`addNS` add namespace definition? Default is `FALSE`

Method `setHref()`**:** Set Href attribute

Usage:

`ISOAbstractObject$setHref(href)`

Arguments:

`href` href

Method `setCodeList()`**:** Set codelist attribute

Usage:

`ISOAbstractObject$setCodeList(codeList)`

Arguments:

`codeList` codelist

Method `setCodeListValue()`**:** Set codelist value

Usage:

`ISOAbstractObject$setCodeListValue(codeListValue)`

Arguments:

`codeListValue` codelist value

Method `setCodeSpace()`**:** Set codeSpace

Usage:

`ISOAbstractObject$setCodeSpace(codeSpace)`

Arguments:

`codeSpace` codespace

Method `setValue()`**:** Set value

Usage:

`ISOAbstractObject$setValue(value)`

Arguments:

`value` value

Method `isDocument()`**:** Util to check where object refers to a emetadata document (eg. ISOMetadata or ISOFeatureCatalogue)

Usage:

`ISOAbstractObject$isDocument()`

Returns: `TRUE` if a document, `FALSE` otherwise

Method `isFieldInheritedFrom()`**:** Indicates the class a field inherits from

Usage:

`ISOAbstractObject$isFieldInheritedFrom(field)`

Arguments:

`field` field

Returns: an object generator of class R6Class

Method `createLocalisedProperty()`**:** Creates a localised property

Usage:

`ISOAbstractObject$createLocalisedProperty(text, locales)`

Arguments:

`text` text

`locales` a list of localized names

Note

Abstract ISO Metadata class used internally by geometa

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

`ISOAbstractPositionalAccuracy` *ISOAbstractPositionalAccuracy*

Description

ISOAbstractPositionalAccuracy

ISOAbstractPositionalAccuracy

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAbstractPositionalAccuracy

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `ISOAbstractPositionalAccuracy`

Methods

Public methods:

- `ISOAbstractPositionalAccuracy$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

ISOAbstractPropertyType *ISOAbstractPropertyType*

Description

ISOAbstractPropertyType

ISOAbstractPropertyType

Format

R6Class object.

Value

Object of R6Class for modelling an ISOAbstractPropertyType

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISOAbstractCarrierOfCharacteristics -> ISOAbstractPropertyType

Public fields

`memberName` typeName [1..1]: ISOLocalName

`definition` definition [0..1]: character

`cardinality` cardinality [1..1]: ISOMultiplicity

`definitionReference` definitionReference [0..1]

`featureCatalogue` featureCatalogue [0..1]

Methods

Public methods:

- `ISOAbstractPropertyType$new()`
- `ISOAbstractPropertyType$setMemberName()`
- `ISOAbstractPropertyType$setDefinition()`
- `ISOAbstractPropertyType$setCardinality()`
- `ISOAbstractPropertyType$setDefinitionReference()`
- `ISOAbstractPropertyType$setFeatureCatalogue()`
- `ISOAbstractPropertyType$clone()`

Method `new()`: Initializes object

Usage:

```
ISOAbstractPropertyType$new(xml = NULL, defaults = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` default values

Method `setMemberName()`: Set member name

Usage:

```
ISOAbstractPropertyType$setMemberName(memberName)
```

Arguments:

`memberName` member name object of class character or ISOLocalName

Method `setDefinition()`: Set definition

Usage:

```
ISOAbstractPropertyType$setDefinition(definition, locales = NULL)
```

Arguments:

`definition` definition

`locales` locale definitions, as list

Method `setCardinality()`: Set cardinality

Usage:

```
ISOAbstractPropertyType$setCardinality(lower, upper)
```

Arguments:

`lower` lower

`upper` upper

Method `setDefinitionReference()`: Set definition reference

Usage:

```
ISOAbstractPropertyType$setDefinitionReference(definitionReference)
```

Arguments:

`definitionReference` object of class ISODefinitionReference

Method `setFeatureCatalogue()`: Set feature catalogue

Usage:

```
ISOAbstractPropertyType$setFeatureCatalogue(featureCatalogue)
```

Arguments:

`featureCatalogue` object of class ISOFeatureCatalogue

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

`ISOAbstractReferenceSystem` *ISOAbstractReferenceSystem*

Description

ISOAbstractReferenceSystem

ISOAbstractReferenceSystem

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO abstract RS Reference system

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOAbstractReferenceSystem`

Public fields

`name` name

`domainOfValidity` domain of validity

Methods

Public methods:

- `ISOAbstractReferenceSystem$new()`
- `ISOAbstractReferenceSystem$setName()`
- `ISOAbstractReferenceSystem$addDomainOfValidity()`
- `ISOAbstractReferenceSystem$delDomainOfValidity()`
- `ISOAbstractReferenceSystem$clone()`

Method `new()`: Initializes object

Usage:

```
ISOAbstractReferenceSystem$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setName()`**:** Set name

Usage:

```
ISOAbstractReferenceSystem$setName(name)
```

Arguments:

`name` name, object of class ISOReferenceIdentifier

Method `addDomainOfValidity()`**:** Adds domain of validity

Usage:

```
ISOAbstractReferenceSystem$addDomainOfValidity(domainOfValidity)
```

Arguments:

`domainOfValidity` object of class ISOExtent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDomainOfValidity()`**:** Deletes domain of validity

Usage:

```
ISOAbstractReferenceSystem$delDomainOfValidity(domainOfValidity)
```

Arguments:

`domainOfValidity` object of class ISOExtent

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

abstract class

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

ISOAbstractResult *ISOAbstractResult*

Description

ISOAbstractResult

ISOAbstractResult

Format

R6Class object.

Value

Object of R6Class for modelling an ISO Result

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> ISOAbstractResult

Public fields

`specification` specification

`explanation` explanation

`pass` pass

Methods

Public methods:

- `ISOAbstractResult$new()`
- `ISOAbstractResult$clone()`

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

Usage:

```
ISOAbstractResult$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

abstract class

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOAbstractTemporalAccuracy` *ISOAbstractTemporalAccuracy*

Description

ISOAbstractTemporalAccuracy

ISOAbstractTemporalAccuracy

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAbstractTemporalAccuracy

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `ISOAbstractTemporalAccuracy`

Methods

Public methods:

- `ISOAbstractTemporalAccuracy$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOAbstractThematicAccuracy` *ISOAbstractThematicAccuracy*

Description

ISOAbstractThematicAccuracy

ISOAbstractThematicAccuracy

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAbstractThematicAccuracy

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `ISOAbstractThematicAccuracy`

Methods

Public methods:

- `ISOAbstractThematicAccuracy$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOAccuracyOfATimeMeasurement` *ISOAccuracyOfATimeMeasurement*

Description

ISOAccuracyOfATimeMeasurement

ISOAccuracyOfATimeMeasurement

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAccuracyOfATimeMeasurement

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `geometa::ISOAbstractTemporalAccuracy` -> `ISOAccuracyOfATimeMeasurement`

Methods

Public methods:

- `ISOAccuracyOfATimeMeasurement$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOAccuracyOfATimeMeasurement$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

`ISOAddress` *ISOAddress*

Description

ISOAddress

ISOAddress

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Address

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOAddress`

Public fields

`deliveryPoint` delivery point

`city` city

`postalCode` postal code

`country` country

`electronicMailAddress` email

Methods

Public methods:

- `ISOAddress$new()`
- `ISOAddress$setDeliveryPoint()`
- `ISOAddress$setCity()`
- `ISOAddress$setPostalCode()`
- `ISOAddress$setCountry()`
- `ISOAddress$setEmail()`
- `ISOAddress$clone()`

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

Usage:

```
ISOAddress$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setDeliveryPoint()`**:** Set delivery point

Usage:

```
ISOAddress$setDeliveryPoint(deliveryPoint, locales = NULL)
```

Arguments:

`deliveryPoint` delivery point

`locales` list of localized names

Method `setCity()`**:** Set city

Usage:

```
ISOAddress$setCity(city, locales = NULL)
```

Arguments:

`city` city

`locales` list of localized names

Method `setPostalCode()`**:** Set postal code

Usage:

```
ISOAddress$setPostalCode(postalCode, locales = NULL)
```

Arguments:

`postalCode` postal code

`locales` list of localized names

Method `setCountry()`**:** Set country

Usage:

```
ISOAddress$setCountry(country, locales = NULL)
```

Arguments:

`country` country

`locales` list of localized names

Method `setEmail()`**:** Set email

Usage:

```
ISOAddress$setEmail(email, locales = NULL)
```

Arguments:

`email` email

`locales` list of localized names

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOAddress$new()
md$setDeliveryPoint("theaddress")
md$setCity("thecity")
md$setPostalCode("111")
md$setCountry("France")
md$setEmail("someone@theorg.org")
xml <- md$encode()
```

`ISOAggregateInformation` *ISOAggregateInformation*

Description

ISOAggregateInformation

ISOAggregateInformation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling a ISO AggregateInformation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOAggregateInformation`

Public fields

`aggregateDataSetName` aggregate dataset name

`aggregateDataSetIdentifier` aggregate dataset identifier

`associationType` association type

`initiativeType` initiative type

Methods

Public methods:

- `ISOAggregateInformation$new()`
- `ISOAggregateInformation$setAggregateDataSetName()`
- `ISOAggregateInformation$setAggregateDataSetIdentifier()`
- `ISOAggregateInformation$setAssociationType()`
- `ISOAggregateInformation$setInitiativeType()`
- `ISOAggregateInformation$clone()`

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

Usage:

```
ISOAggregateInformation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setAggregateDataSetName()`**:** Set aggregate dataset name

Usage:

```
ISOAggregateInformation$setAggregateDataSetName(datasetName)
```

Arguments:

`datasetName` object of class ISOCitation

Method `setAggregateDataSetIdentifier()`**:** Set aggregate dataset identifier

Usage:

```
ISOAggregateInformation$setAggregateDataSetIdentifier(datasetIdentifier)
```

Arguments:

`datasetIdentifier` object of class ISOMetaIdentifier

Method `setAssociationType()`**:** Set association type

Usage:

```
ISOAggregateInformation$setAssociationType(associationType)
```

Arguments:

`associationType` object of class ISOAssociationType or character value among values from `ISOAssociationType$values()`

Method `setInitiativeType()`**:** Set association type

Usage:

```
ISOAggregateInformation$setInitiativeType(initiativeType)
```

Arguments:

`initiativeType` object of class ISOInitiativeType or character value among values from `ISOInitiativeType$values(`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
md <- ISOAggregateInformation$new()

#adding a point of contact
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
rp$setOrganisationName("somewhere")
rp$setPositionName("someposition")
rp$setRole("pointOfContact")
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact$setOnlineResource(res)
rp$setContactInfo(contact)
#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(ISOdate(2015,1,1))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp)
md$setAggregateDataSetName(ct)

md$setAssociationType("source")
md$setInitiativeType("investigation")

xml <- md$encode()
```

`ISOAnchor` *ISOAnchor*

Description

ISOAnchor

ISOAnchor

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Anchor

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOAnchor`

Methods

Public methods:

- `ISOAnchor$new()`
- `ISOAnchor$clone()`

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

Usage:

`ISOAnchor$new(xml = NULL, name = NULL, ...)`

Arguments:

`xml` object of class XMLInternalNode-class

`name` name

`...` attributes for XML encoding

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

Usage:

`ISOAnchor$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19139:2007 Geographic information – XML

Examples

```
md <- ISOAnchor$new(name = "some entity name", href = "someentityuri")
xml <- md$encode()
```

`ISOAngle` *ISOAngle*

Description

ISOAngle

ISOAngle

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAngle measure

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOMeasure` -> `ISOAngle`

Methods

Public methods:

- `ISOAngle$new()`
- `ISOAngle$clone()`

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

Usage:

```
ISOAngle$new(xml = NULL, value, uom, useUomURI = FALSE)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`uom` uom symbol of unit of measure used

`useUomURI` use uom URI. Default is `FALSE`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOApplicationSchemaInformation` *ISOApplicationSchemaInformation*

Description

ISOApplicationSchemaInformation

ISOApplicationSchemaInformation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO ApplicationSchemaInformation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOApplicationSchemaInformation`

Public fields

`name` name [1..1]

`schemaLanguage` chemaLanguage [1..1]

`constraintLanguage` constraintLanguage [1..1]

`schemaAscii` schemaAscii [0..1]

`graphicsFile` graphicsFile [0..1]

`softwareDevelopmentFile` softwareDevelopmentFile [0..1]

`softwareDevelopmentFileFormat` softwareDevelopmentFileFormat [0..1]

Methods

Public methods:

- `ISOApplicationSchemaInformation$new()`
- `ISOApplicationSchemaInformation$setName()`
- `ISOApplicationSchemaInformation$setSchemaLanguage()`
- `ISOApplicationSchemaInformation$setConstraintLanguage()`
- `ISOApplicationSchemaInformation$setSchemaAscii()`
- `ISOApplicationSchemaInformation$setGraphicsFile()`
- `ISOApplicationSchemaInformation$setSoftwareDevelopmentFile()`

- `ISOApplicationSchemaInformation$setSoftwareDevelopmentFileFormat()`
- `ISOApplicationSchemaInformation$clone()`

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

Usage:

`ISOApplicationSchemaInformation$new(xml = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

Method `setName()`**:** Set name

Usage:

`ISOApplicationSchemaInformation$setName(name)`

Arguments:

`name` name

Method `setSchemaLanguage()`**:** Set schema language

Usage:

`ISOApplicationSchemaInformation$setSchemaLanguage(schemaLanguage)`

Arguments:

`schemaLanguage` schema language

Method `setConstraintLanguage()`**:** Set constraint language

Usage:

`ISOApplicationSchemaInformation$setConstraintLanguage(constraintLanguage)`

Arguments:

`constraintLanguage` constraint language

Method `setSchemaAscii()`**:** Set schema Ascii

Usage:

`ISOApplicationSchemaInformation$setSchemaAscii(schemaAscii)`

Arguments:

`schemaAscii` schema Ascii

Method `setGraphicsFile()`**:** Set graphics file

Usage:

`ISOApplicationSchemaInformation$setGraphicsFile(graphicsFile)`

Arguments:

`graphicsFile` graphics file

Method `setSoftwareDevelopmentFile()`**:** Set software development file

Usage:

`ISOApplicationSchemaInformation$setSoftwareDevelopmentFile(file)`

Arguments:

`file` file

Method `setSoftwareDevelopmentFileFormat():` Set software development file format

Usage:

```
ISOApplicationSchemaInformation$setSoftwareDevelopmentFileFormat(format)
```

Arguments:

`format` file format

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOAssociation` *ISOAssociation*

Description

ISOAssociation

ISOAssociation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAssociation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOAssociation`

Methods

Public methods:

- `ISOAssociation$new()`
- `ISOAssociation$clone()`

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

Usage:

```
ISOAssociation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOAssociationRole` *ISOAssociationRole*

Description

ISOAssociationRole

ISOAssociationRole

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOAssociationRole

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractCarrierOfCharacteristics` -> `geometa::ISOAbstractPropertyType` -> `geometa::ISOPropertyType` -> `ISOAssociationRole`

Public fields

`type` type: ISORoleType

`isOrdered` isOrdered: logical

`isNavigable` isNavigable: logical

`relation` relation: ISOAssociationRole

`rolePlayer` rolePlayer: ISOFeatureType

Methods

Public methods:

- `ISOAssociationRole$new()`
- `ISOAssociationRole$setRoleType()`
- `ISOAssociationRole$setIsOrdered()`
- `ISOAssociationRole$setIsNavigable()`
- `ISOAssociationRole$setRelation()`
- `ISOAssociationRole$addRolePlayer()`
- `ISOAssociationRole$delRolePlayer()`
- `ISOAssociationRole$clone()`

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

Usage:

```
ISOAssociationRole$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setRoleType()`**:** Set role type

Usage:

```
ISOAssociationRole$setRoleType(roleType)
```

Arguments:

`roleType` role type, object of class ISORoleType or any character among values returned by `ISORoleType$values()`

Method `setIsOrdered()`**:** Set is ordered

Usage:

```
ISOAssociationRole$setIsOrdered(isOrdered)
```

Arguments:

`isOrdered` object of class logical

Method `setIsNavigable()`**:** Set is navigable

Usage:

```
ISOAssociationRole$setIsNavigable(isNavigable)
```

Arguments:

`isNavigable` object of class logical

Method `setRelation()`**:** Set relation

Usage:

```
ISOAssociationRole$setRelation(relation)
```

Arguments:

`relation` relation

Method `addRolePlayer()`**:** Adds role player

Usage:

```
ISOAssociationRole$addRolePlayer(rolePlayer)
```

Arguments:

`rolePlayer` object of class ISOFeatureType

Returns: `TRUE` if added, `FALSE` otherwise

Method `delRolePlayer()`**:** Deletes role player

Usage:

```
ISOAssociationRole$delRolePlayer(rolePlayer)
```

Arguments:

`rolePlayer` object of class ISOFeatureType

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

ISOAssociationType *ISOAssociationType*

Description

ISOAssociationType

ISOAssociationType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO AssociationType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOAssociationType`

Methods

Public methods:

- `ISOAssociationType$new()`
- `ISOAssociationType$clone()`

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

Usage:

```
ISOAssociationType$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOAssociationType$values(labels = TRUE)

#geomOnly
geomOnly <- ISOAssociationType$new(value = "source")
```

`ISOAttributes` *ISOAttributes*

Description

ISOAttributes

ISOAttributes

Format

`R6Class` object.

Value

Spatial object of `R6Class` for modelling a list of ISO xml attributes

Public fields

`attrs` attrs

Methods

Public methods:

- `ISOAttributes$new()`
- `ISOAttributes$clone()`

Method `new()`**:** method is used to instantiate a vector of attributes to be used for empty element properties.

Usage:

`ISOAttributes$new(...)`

Arguments:

`...` list of attributes

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

Usage:

`ISOAttributes$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```
attrs <- ISOAttributes$new(href = "http://somelink", title = "sometitle")
```

`ISOBand` *ISOBand*

Description

ISOBand

ISOBand

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOBand

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISORangeDimension` -> `ISOBand`

Public fields

`maxValue` maxValue [0..1] : numeric

`minValue` minValue [0..1] : numeric

`units` units [0..1] : GMLUnitDefinition

`peakResponse` peakResponse [0..1] : numeric

`bitsPerValue` bitsPerValue [0..1] : integer

`toneGradation` toneGradation [0..1] : integer

`scaleFactor` scaleFactor [0..1] : numeric

`offset` offset [0..1] : numeric

Methods

Public methods:

- `ISOBand$new()`
- `ISOBand$setMaxValue()`
- `ISOBand$setMinValue()`
- `ISOBand$setUnits()`
- `ISOBand$setPeakResponse()`
- `ISOBand$setBitsPerValue()`
- `ISOBand$setToneGradation()`
- `ISOBand$setScaleFactor()`
- `ISOBand$setOffset()`
- `ISOBand$clone()`

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

Usage:

```
ISOBand$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setMaxValue()`**:** Set max value

Usage:

```
ISOBand$setMaxValue(maxValue)
```

Arguments:

`maxValue` max value, object of class numeric

Method `setMinValue()`**:** Set min value

Usage:

```
ISOBand$setMinValue(minValue)
```

Arguments:

`minValue` min value, object of class numeric

Method `setUnits()`**:** Set unit definition

Usage:

```
ISOBand$setUnits(uom)
```

Arguments:

`uom` object of class GMLUnitDefinition

Method `setPeakResponse()`**:** Set peak response

Usage:

```
ISOBand$setPeakResponse(peakResponse)
```

Arguments:

`peakResponse` object of class numeric

Method `setBitsPerValue()`: Set bits per value

Usage:

`ISOBand$setBitsPerValue(bitsPerValue)`

Arguments:

`bitsPerValue` object of class numeric

Method `setToneGradation()`: Set tone gradation

Usage:

`ISOBand$setToneGradation(toneGradation)`

Arguments:

`toneGradation` object of class numeric

Method `setScaleFactor()`: Set scale factor

Usage:

`ISOBand$setScaleFactor(scaleFactor)`

Arguments:

`scaleFactor` object of class numeric

Method `setOffset()`: Set offset

Usage:

`ISOBand$setOffset(offset)`

Arguments:

`offset` object of class numeric

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

Usage:

`ISOBand$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```
#create band range dimension
md <- ISOBand$new()
md$setSequenceIdentifier(ISOMemberName$new(aName = "name", attributeType = "type"))
md$setDescriptor("descriptor")
md$setMaxValue(10)
md$setMinValue(1)
gml <- GMLBaseUnit$new(id = "ID")
gml$setDescriptionReference("someref")
gml$setIdentifier("identifier", "codespace")
```

```
gml$addName("name1", "codespace")
gml$addName("name2", "codespace")
gml$setQuantityTypeReference("someref")
gml$setCatalogSymbol("symbol")
gml$setUnitsSystem("somelink")
md$setUnits(gml)
md$setPeakResponse(9)
md$setBitsPerValue(5)
md$setToneGradation(100)
md$setScaleFactor(1)
md$setOffset(4)
xml <- md$encode()
```

ISOBaseBoolean *ISOBaseBoolean*

Description

ISOBaseBoolean

ISOBaseBoolean

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Boolean

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOBaseBoolean`

Public fields

`value` value

Methods

Public methods:

- `ISOBaseBoolean$new()`
- `ISOBaseBoolean$clone()`

Method `new()`**:** Initializes a base boolean object

Usage:

`ISOBaseBoolean$new(xml = NULL, value)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOBaseCharacterString` *ISOBaseCharacterString*

Description

ISOBaseCharacterString

ISOBaseCharacterString

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO BaseCharacterString

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOBaseCharacterString`

Public fields

`value` value

Methods

Public methods:

- `ISOBaseCharacterString$new()`
- `ISOBaseCharacterString$clone()`

Method `new()`**:** Initializes a base character object

Usage:

```
ISOBaseCharacterString$new(xml = NULL, value)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOBaseDate` *ISOBaseDate*

Description

ISOBaseDate

ISOBaseDate

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Date

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOBaseDate`

Public fields

`value` value

Methods

Public methods:

- `ISOBaseDate$new()`
- `ISOBaseDate$clone()`

Method `new()`**:** Initializes a base date object

Usage:

```
ISOBaseDate$new(xml = NULL, value = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

`ISOBaseDateTime` *ISOBaseDateTime*

Description

ISOBaseDateTime

ISOBaseDateTime

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO DateTime

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOBaseDateTime`

Public fields

`value` value

Methods

Public methods:

- `ISOBaseDateTime$new()`
- `ISOBaseDateTime$clone()`

Method `new()`**:** Initializes a base datetime object

Usage:

```
ISOBaseDateTime$new(xml = NULL, value = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOBaseDecimal` *ISOBaseDecimal*

Description

ISOBaseDecimal

ISOBaseDecimal

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Decimal

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOBaseDecimal`

Public fields

`value` value

Methods

Public methods:

- `ISOBaseDecimal$new()`
- `ISOBaseDecimal$clone()`

Method `new()`**:** Initializes a base decimal object

Usage:

```
ISOBaseDecimal$new(xml = NULL, value)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOBaseInteger` *ISOBaseInteger*

Description

ISOBaseInteger

ISOBaseInteger

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Integer

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOBaseInteger`

Public fields

`value` value

Methods

Public methods:

- `ISOBaseInteger$new()`
- `ISOBaseInteger$clone()`

Method `new()`**:** Initializes a base integer object

Usage:

```
ISOBaseInteger$new(xml = NULL, value)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOBaseReal` *ISOBaseReal*

Description

ISOBaseReal

ISOBaseReal

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Real

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOBaseReal`

Public fields

`value` value

Methods

Public methods:

- `ISOBaseReal$new()`
- `ISOBaseReal$clone()`

Method `new()`**:** Initializes a base real object

Usage:

```
ISOBaseReal$new(xml = NULL, value)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

`ISOBinary` *ISOBinary*

Description

ISOBinary

ISOBinary

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO UnlimitedInteger

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOBinary`

Public fields

`value` value

`attrs` attrs

Methods

Public methods:

- `ISOBinary$new()`
- `ISOBinary$clone()`

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

Usage:

```
ISOBinary$new(xml = NULL, value)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

Examples

```
bin <- ISOBinary$new(value = "http://someuri")
```

`ISOBinding` *ISOBinding*

Description

ISOBinding

ISOBinding

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOBinding

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractCarrierOfCharacteristics` -> `ISOBinding`

Public fields

`description` description [0..1]: character

`globalProperty` globalProperty [1..1]: ISOPropertyType

Methods

Public methods:

- `ISOBinding$setDescription()`
- `ISOBinding$setPropertyType()`
- `ISOBinding$clone()`

Method `setDescription()`**:** Set description

Usage:

`ISOBinding$setDescription(description, locales = NULL)`

Arguments:

`description` description

`locales` list of localized descriptions

Method `setPropertyType()`**:** Set property type.

Usage:

`ISOBinding$setPropertyType(propertyType)`

Arguments:

`propertyType` property type, object of class ISOPropertyType

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

Usage:

`ISOBinding$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

`ISOBoundAssociationRole` *ISOBoundAssociationRole*

Description

ISOBoundAssociationRole

ISOBoundAssociationRole

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOBoundAssociationRole

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractCarrierOfCharacteristics` -> `geometa::ISOBinding` -> `ISOBoundAssociationRole`

Public fields

`rolePlayer` rolePlayer [0..1]: ISOFeatureType

Methods

Public methods:

- `ISOBoundAssociationRole$setRolePlayer()`
- `ISOBoundAssociationRole$clone()`

Method `setRolePlayer()`**:** set role player

Usage:

```
ISOBoundAssociationRole$setRolePlayer(rolePlayer)
```

Arguments:

`rolePlayer` object of class ISOFeatureType

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

`ISOBoundFeatureAttribute` *ISOBoundFeatureAttribute*

Description

ISOBoundFeatureAttribute

ISOBoundFeatureAttribute

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOBoundFeatureAttribute

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractCarrierOfCharacteristics` -> `geometa::ISOBinding` -> `ISOBoundFeatureAttribute`

Public fields

`valueType` valueType [0..1]: ISOTypeName

Methods

Public methods:

- `ISOBoundFeatureAttribute$setTypeName()`
- `ISOBoundFeatureAttribute$clone()`

Method `setTypeName()`**:** Set type name

Usage:

`ISOBoundFeatureAttribute$setTypeName(typeName)`

Arguments:

`typeName` object of class ISOTypeName or character

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

Usage:

`ISOBoundFeatureAttribute$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

`ISOBoundingPolygon` *ISOBoundingPolygon*

Description

ISOBoundingPolygon

ISOBoundingPolygon

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO BoundingPolygon

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOGeographicExtent` -> `ISOBoundingPolygon`

Public fields

`polygon` list of polygons

Methods

Public methods:

- `ISOBoundingPolygon$new()`
- `ISOBoundingPolygon$addPolygon()`
- `ISOBoundingPolygon$delPolygon()`
- `ISOBoundingPolygon$clone()`

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

Usage:

```
ISOBoundingPolygon$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addPolygon()`**:** Adds polygon

Usage:

```
ISOBoundingPolygon$addPolygon(x)
```

Arguments:

`x` geometry object from **sf** or object of class inheriting GMLAbstractGeometry

Returns: `TRUE` if added, `FALSE` otherwise

Method `delPolygon()`**:** Deletes polygon

Usage:

```
ISOBoundingPolygon$delPolygon(x)
```

Arguments:

`x` geometry object from **sf** or object of class inheriting GMLAbstractGeometry

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Experimental

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOBrowseGraphic` *ISOBrowseGraphic*

Description

ISOBrowseGraphic

ISOBrowseGraphic

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO BrowseGraphic

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOBrowseGraphic`

Public fields

`fileName` file name

`fileDescription` file description

`fileType` file type

Methods

Public methods:

- `ISOBrowseGraphic$new()`
- `ISOBrowseGraphic$setFileName()`
- `ISOBrowseGraphic$setFileDescription()`
- `ISOBrowseGraphic$setFileType()`
- `ISOBrowseGraphic$clone()`

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

Usage:

```
ISOBrowseGraphic$new(
  xml = NULL,
  fileName = NULL,
  fileDescription = NULL,
  fileType = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`fileName` file name

`fileDescription` file description

`fileType` file type

Method `setFileName()`**:** Set file name

Usage:

```
ISOBrowseGraphic$setFileName(fileName, locales = NULL)
```

Arguments:

`fileName` file name

`locales` a list of localized names. Default is `NULL`

Method `setFileDescription()`**:** Set file description

Usage:

```
ISOBrowseGraphic$setFileDescription(fileDescription, locales = NULL)
```

Arguments:

`fileDescription` file description

`locales` a list of localized descriptions. Default is `NULL`

Method `setFileType()`**:** Set file type

Usage:

```
ISOBrowseGraphic$setFileType(fileType, locales = NULL)
```

Arguments:

`fileType` file type

`locales` a list of localized types. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOBrowseGraphic$new(
 fileName = "http://wwww.somefile.org/png",
 fileDescription = "Map Overview",
 fileType = "image/png"
)
xml <- md$encode()
```

`ISOCarrierOfCharacteristics` *ISOCarrierOfCharacteristics*

Description

ISOCarrierOfCharacteristics

ISOCarrierOfCharacteristics

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOCarrierOfCharacteristics

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractCarrierOfCharacteristics` -> `ISOCarrierOfCharacteristics`

Methods

Public methods:

- `ISOCarrierOfCharacteristics$new()`
- `ISOCarrierOfCharacteristics$clone()`

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

Usage:

`ISOCarrierOfCharacteristics$new(xml = NULL, defaults = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` defaults

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

`ISOCellGeometry` *ISOCellGeometry*

Description

ISOCellGeometry

ISOCellGeometry

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO CellGeometryCode

Methods

`new(xml,value, description)` This method is used to instantiate an `ISOCellGeometry`

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOCellGeometry`

Methods

Public methods:

- `ISOCellGeometry$new()`
- `ISOCellGeometry$clone()`

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

Usage:

`ISOCellGeometry$new(xml = NULL, value, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOCellGeometry$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOCellGeometry$values(labels = TRUE)

#example of 'point' cell geometry code
pointCode <- ISOCellGeometry$new(value = "point")
```

`ISOCharacterSet` *ISOCharacterSet*

Description

ISOCharacterSet

ISOCharacterSet

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO CharacterSet

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOCharacterSet`

Methods

Public methods:

- `ISOCharacterSet$new()`
- `ISOCharacterSet$clone()`

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

Usage:

`ISOCharacterSet$new(xml = NULL, value, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOCharacterSet$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOCharacterSet$values(labels = TRUE)

#some charset
charset <- ISOCharacterSet$new(value = "utf8")
```

`ISOCitation` *ISOCitation*

Description

ISOCitation

ISOCitation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Citation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOCitation`

Public fields

`title` title

`alternateTitle` alternate title

`date` date list

`edition` edition

`editionDate` edition date

`identifier` identifier list

`citedResponsibleParty` list of cited responsible parties

`presentationForm` list of presentation forms

`series` series

`otherCitationDetails` other citation details

`collectiveTitle` collective title

`ISBN` ISBN

`ISSN` ISSN

Methods

Public methods:

- `ISOCitation$new()`
- `ISOCitation$setTitle()`
- `ISOCitation$setAlternateTitle()`
- `ISOCitation$addAlternateTitle()`

- `ISOCitation$delAlternateTitle()`
- `ISOCitation$addDate()`
- `ISOCitation$setEdition()`
- `ISOCitation$setEditionDate()`
- `ISOCitation$setIdentifier()`
- `ISOCitation$addIdentifier()`
- `ISOCitation$delIdentifier()`
- `ISOCitation$setCitedResponsibleParty()`
- `ISOCitation$addCitedResponsibleParty()`
- `ISOCitation$delCitedResponsibleParty()`
- `ISOCitation$setPresentationForm()`
- `ISOCitation$addPresentationForm()`
- `ISOCitation$delPresentationForm()`
- `ISOCitation$setSeries()`
- `ISOCitation$setOtherCitationDetails()`
- `ISOCitation$setCollectiveTitle()`
- `ISOCitation$setISBN()`
- `ISOCitation$setISSN()`
- `ISOCitation$clone()`

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

Usage:

```
ISOCitation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setTitle()`**:** Set title

Usage:

```
ISOCitation$setTitle(title, locales = NULL)
```

Arguments:

`title` title

`locales` list of localized names. Default is `NULL`

Method `setAlternateTitle()`**:** Set alternate title

Usage:

```
ISOCitation$setAlternateTitle(alternateTitle, locales = NULL)
```

Arguments:

`alternateTitle` alternate title

`locales` list of localized names. Default is `NULL`

Method `addAlternateTitle()`**:** Adds alternate title

Usage:

```
ISOCitation$addAlternateTitle(alternateTitle, locales = NULL)
```

Arguments:

`alternateTitle` alternate title

`locales` list of localized titles. Default is `NULL`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delAlternateTitle()`**:** Deletes alternate title

Usage:

```
ISOCitation$delAlternateTitle(alternateTitle, locales = NULL)
```

Arguments:

`alternateTitle` alternate title

`locales` list of localized titles. Default is `NULL`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addDate()`**:** Adds date

Usage:

```
ISOCitation$addDate(date)
```

Arguments:

`date` date

Returns: `TRUE` if added, `FALSE` otherwise

Method `setEdition()`**:** Set edition

Usage:

```
ISOCitation$setEdition(edition)
```

Arguments:

`edition` edition

Method `setEditionDate()`**:** Sets the edition date, either an ISODate object containing date and dateType or a simple R date "POSIXct"/"POSIXt" object. For thesaurus citations, an ISODate should be used while for the general citation of `ISODataIdentification`, a simple R date should be used.

Usage:

```
ISOCitation$setEditionDate(editionDate)
```

Arguments:

`editionDate` object of class Date or POSIXct

Method `setIdentifier()`**:** Set identifier

Usage:

```
ISOCitation$setIdentifier(identifier)
```

Arguments:

`identifier` identifier, object of class ISOMetaIdentifier

Method `addIdentifier()`**:** Adds identifier

Usage:

```
ISOCitation$addIdentifier(identifier)
```

Arguments:

`identifier` identifier, object of class ISOMetaIdentifier

`locales` list of localized identifiers. Default is `NULL`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delIdentifier()`**:** Deletes identifier

Usage:

```
ISOCitation$delIdentifier(identifier)
```

Arguments:

`identifier` identifier, object of class ISOMetaIdentifier

`locales` list of localized identifiers. Default is `NULL`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setCitedResponsibleParty()`**:** Set cited responsible party

Usage:

```
ISOCitation$setCitedResponsibleParty(rp)
```

Arguments:

`rp` cited responsible party, object of class ISOResponsibleParty

Method `addCitedResponsibleParty()`**:** Adds cited responsible party

Usage:

```
ISOCitation$addCitedResponsibleParty(rp)
```

Arguments:

`rp` cited responsible party, object of class ISOResponsibleParty

`locales` list of localized responsible parties. Default is `NULL`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delCitedResponsibleParty()`**:** Deletes cited responsible party

Usage:

```
ISOCitation$delCitedResponsibleParty(rp)
```

Arguments:

`rp` cited responsible party, object of class ISOResponsibleParty

`locales` list of localized responsible parties. Default is `NULL`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setPresentationForm()`**:** Sets presentation form

Usage:

```
ISOCitation$setPresentationForm(presentationForm)
```

Arguments:

`presentationForm` presentation form, object of class ISOPresentationForm or character among values returned by `ISOPresentationForm$values()`

Method `addPresentationForm()`**:** Adds presentation form

Usage:

```
ISOCitation$addPresentationForm(presentationForm)
```

Arguments:

`presentationForm` presentation form, object of class ISOPresentationForm or character among values returned by `ISOPresentationForm$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delPresentationForm()`**:** Deletes presentation form

Usage:

```
ISOCitation$delPresentationForm(presentationForm)
```

Arguments:

`presentationForm` presentation form, object of class ISOPresentationForm or character among values returned by `ISOPresentationForm$values()`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setSeries()`**:** Set series

Usage:

```
ISOCitation$setSeries(series)
```

Arguments:

`series` object of class ISOCitationSeries

Method `setOtherCitationDetails()`**:** Set other citation details

Usage:

```
ISOCitation$setOtherCitationDetails(otherCitationDetails, locales = NULL)
```

Arguments:

`otherCitationDetails` other citation details

`locales` list of localized other citation details. Default is `NULL`

Method `setCollectiveTitle()`**:** Set collective title

Usage:

```
ISOCitation$setCollectiveTitle(collectiveTitle, locales = NULL)
```

Arguments:

`collectiveTitle` collective title

`locales` list of localized titles. Default is `NULL`

Method `setISBN()`**:** Set ISBN

Usage:

```
ISOCitation$setISBN(isbn)
```

Arguments:

`isbn` isbn

Method `setISSN()`**:** Set ISSN

Usage:

```
ISOCitation$setISSN(issn)
```

Arguments:

`issn` issn

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create ISOCitation
md <- ISOCitation$new()
md$setTitle("sometitle")
md$setEdition("1.0")
md$setEditionDate(ISOdate(2015,1,1))
md$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
md$addPresentationForm("mapDigital")

#add a cited responsible party
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
rp$setOrganisationName("somewhere")
rp$setPositionName("someposition")
rp$setRole("pointOfContact")
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
```

```
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact$setOnlineResource(res)
rp$setContactInfo(contact)
md$addCitedResponsibleParty(rp)
xml <- md$encode()
```

`ISOCitationSeries` *ISOCitationSeries*

Description

ISOCitationSeries

ISOCitationSeries

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOCitationSeries

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOCitationSeries`

Public fields

`name` name [0..1]

`issueIdentification` issueIdentification [0..1]

`page` page [0..1]

Methods

Public methods:

- `ISOCitationSeries$new()`
- `ISOCitationSeries$setName()`
- `ISOCitationSeries$setIssueIdentification()`
- `ISOCitationSeries$setPage()`
- `ISOCitationSeries$clone()`

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

Usage:

```
ISOCitationSeries$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setName()`**:** Set name

Usage:

```
ISOCitationSeries$setName(name, locales = NULL)
```

Arguments:

`name` name

`locales` list of localized names. Default is `NULL`

Method `setIssueIdentification()`**:** Set issue ID

Usage:

```
ISOCitationSeries$setIssueIdentification(issueId, locales = NULL)
```

Arguments:

`issueId` issueId

`locales` list of localized ids Default is `NULL`

Method `setPage()`**:** Set page

Usage:

```
ISOCitationSeries$setPage(page, locales = NULL)
```

Arguments:

`page` page

`locales` list of localized pages. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOClassification` *ISOClassification*

Description

ISOClassification

ISOClassification

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Classification

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOClassification`

Methods

Public methods:

- `ISOClassification$new()`
- `ISOClassification$clone()`

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

Usage:

```
ISOClassification$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOClassification$values(labels = TRUE)

#restricted classification
cl <- ISOClassification$new(value = "restricted")
```

`ISOCodelist` *ISOCodelist*

Description

ISOCodelist

ISOCodelist

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO codelist

Public fields

`id` id

`refFile` ref file

`codeSpace` code space

`identifier` identifier

`description` description

`entries` entries

Methods

Public methods:

- `ISOCodelist$new()`
- `ISOCodelist$parse()`
- `ISOCodelist$clone()`

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

Usage:

`ISOCodelist$new(refFile, id)`

Arguments:

`refFile` ref file

`id` id

Method `parse()`**:** Parse codelist

Usage:

`ISOCodelist$parse(refFile, id)`

Arguments:

`refFile` ref file

`id` id

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

Usage:

`ISOCodelist$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal codelist XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

`ISOCodeListValue` *ISOCodeListValue*

Description

ISOCodeListValue

ISOCodeListValue

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Metadata codelist element

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOCodeListValue`

Public fields

`codelistId` codelist ID

`attrs` attrs

`value` value

`valueDescription` value description

Methods

Public methods:

- `ISOCodeListValue$new()`
- `ISOCodeListValue$getAcceptedValues()`
- `ISOCodeListValue$clone()`

Method `new()`**:** Method used to instantiate an ISOCodeListValue. By default, `addCodeListAttrs` `= TRUE`, to add codelist atributes to root XML. The parameter `addCodeSpaceAttr = TRUE` by default, and ignored if the valueof `addCodeLisAttrs` is set to `FALSE`. The argument `setValue` sets the value as node text (defaut is `TRUE`). The argument `setValueDescription` allows to force having description set as value, default is `FALSE` in which case the name will be preferred, and in case no name is provided, code value will be used.

Usage:

```
ISOCodeListValue$new(
  xml = NULL,
  id,
  value = NULL,
  description = NULL,
  addCodeListAttrs = TRUE,
  addCodeSpaceAttr = TRUE,
  setValue = TRUE,
  setValueDescription = FALSE
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`id` id

`value` value

`description` description

`addCodeListAttrs` add codelist attributes?

`addCodeSpaceAttr` add codespace attribute?

`setValue` set value?

`setValueDescription` set value description?

Method `getAcceptedValues()`**:** Get accepted values

Usage:

```
ISOCodeListValue$getAcceptedValues()
```

Returns: a vector of class character

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Abstract ISO codelist class used internally by geometa

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOCompletenessCommission` *ISOCompletenessCommission*

Description

ISOCompletenessCommission

ISOCompletenessCommission

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOCompletenessCommission

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `geometa::ISOAbstractThematicAccuracy` -> `ISOCompletenessCommission`

Methods

Public methods:

- `ISOCompletenessCommission$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOCompletenessCommission$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

ISOCompletenessOmission *ISOCompletenessOmission*

Description

ISOCompletenessOmission

ISOCompletenessOmission

Format

R6Class object.

Value

Object of R6Class for modelling an ISOCompletenessOmission

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISODataQualityAbstractElement -> geometa::ISOAbstractThematicAccuracy -> ISOCompletenessOmission

Methods

Public methods:

- ISOCompletenessOmission$clone()

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOCompletenessOmission$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

`ISOConceptualConsistency` *ISOConceptualConsistency*

Description

ISOConceptualConsistency

ISOConceptualConsistency

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOConceptualConsistency

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `geometa::ISOAbstractLogicalConsistency` -> `ISOConceptualConsistency`

Methods

Public methods:

- `ISOConceptualConsistency$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOConceptualConsistency$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

`ISOConformanceResult` *ISOConformanceResult*

Description

ISOConformanceResult

ISOConformanceResult

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO ConformanceResult

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractResult` -> `ISOConformanceResult`

Public fields

`specification` specification

`explanation` explanation

`pass` pass

Methods

Public methods:

- `ISOConformanceResult$new()`
- `ISOConformanceResult$setSpecification()`
- `ISOConformanceResult$setExplanation()`
- `ISOConformanceResult$setPass()`
- `ISOConformanceResult$clone()`

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

Usage:

```
ISOConformanceResult$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setSpecification()`**:** Set specification

Usage:

```
ISOConformanceResult$setSpecification(specification)
```

Arguments:

`specification` specification

Method `setExplanation()`**:** Set explanation about the conformance result

Usage:

`ISOConformanceResult$setExplanation(explanation, locales = NULL)`

Arguments:

`explanation` explanation

`locales` list of localized explanations. Default is `NULL`

Method `setPass()`**:** Set wether the conformance passed or not

Usage:

`ISOConformanceResult$setPass(pass)`

Arguments:

`pass` object of class logical

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

Usage:

`ISOConformanceResult$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOConformanceResult$new()
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
md$setSpecification(spec)
md$setExplanation("some explanation about the conformance")
md$setPass(TRUE)
xml <- md$encode()
```

`ISOConstraint` *ISOConstraint*

Description

ISOConstraint

ISOConstraint

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOConstraint

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOConstraint`

Public fields

`description` description: character

Methods

Public methods:

- `ISOConstraint$new()`
- `ISOConstraint$setDescription()`
- `ISOConstraint$clone()`

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

Usage:

```
ISOConstraint$new(xml = NULL, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`description` description

Method `setDescription()`**:** Set description

Usage:

```
ISOConstraint$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` a list of localized descriptions. Defaut is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

Examples

```
md <- ISOConstraint$new(description = "description")
xml <- md$encode()
```

`ISOConstraints` *ISOConstraints*

Description

ISOConstraints

ISOConstraints

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO abstract Constraints

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOConstraints`

Public fields

`useLimitation` useLimitation [0..*]: character

Methods

Public methods:

- `ISOConstraints$new()`
- `ISOConstraints$addUseLimitation()`
- `ISOConstraints$setUseLimitation()`
- `ISOConstraints$delUseLimitation()`
- `ISOConstraints$clone()`

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

Usage:

```
ISOConstraints$new(xml = NULL, defaults = list())
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of default values

Method `addUseLimitation()`**:** Adds a use limitation

Usage:

```
ISOConstraints$addUseLimitation(useLimitation, locales = NULL)
```

Arguments:

`useLimitation` use limitation

`locales` list of localized use limitations. Default is `NULL`

Returns: `TRUE` if added, `FALSE` otherwise

Method `setUseLimitation()`**:** Adds a use limitation

Usage:

```
ISOConstraints$setUseLimitation(useLimitation, locales = NULL)
```

Arguments:

`useLimitation` use limitation

`locales` list of localized use limitations. Default is `NULL`

Method `delUseLimitation()`**:** Deletes a use limitation

Usage:

```
ISOConstraints$delUseLimitation(useLimitation, locales = NULL)
```

Arguments:

`useLimitation` use limitation

`locales` list of localized use limitations. Default is `NULL`

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Abstract ISO class

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOContact` *ISOContact*

Description

ISOContact

ISOContact

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Contact

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOContact`

Public fields

`phone` phone

`address` address

`onlineResource` online resource

Methods

Public methods:

- `ISOContact$new()`
- `ISOContact$setPhone()`
- `ISOContact$setAddress()`
- `ISOContact$setOnlineResource()`
- `ISOContact$clone()`

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

Usage:

`ISOContact$new(xml = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

Method `setPhone()`**:** Set phone

Usage:

`ISOContact$setPhone(phone)`

Arguments:

`phone` object of class ISOTelephone

Method `setAddress()`**:** Set address

Usage:

`ISOContact$setAddress(address)`

Arguments:

`address` object of class ISOAddress

Method `setOnlineResource()`**:** Set online resource

Usage:

`ISOContact$setOnlineResource(onlineResource)`

Arguments:

`onlineResource` online resource, object of class ISOOnlineResource

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

Usage:

`ISOContact$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
md$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
md$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
md$setOnlineResource(res)
xml <- md$encode()
```

`ISOContentInformation` *ISOContentInformation*

Description

ISOContentInformation

ISOContentInformation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOContentInformation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOContentInformation`

Methods

Public methods:

- `ISOContentInformation$new()`
- `ISOContentInformation$clone()`

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

Usage:

```
ISOContentInformation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Abstract class. Used internally by **geometa**

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOCountry` *ISOCountry*

Description

ISOCountry

ISOCountry

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Country

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOCountry`

Methods

Public methods:

- `ISOCountry$new()`
- `ISOCountry$clone()`

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

Usage:

```
ISOCountry$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOCountry$values(labels = TRUE)

#some charset
charset <- ISOCountry$new(value = "utf8")
```

`ISOCoupledResource` *ISOCoupledResource*

Description

ISOCoupledResource

ISOCoupledResource

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOCoupledResource

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOCoupledResource`

Public fields

`operationName` operationName [1..1]: character

`identifier` identifier [1..1]: character

Methods

Public methods:

- `ISOCoupledResource$new()`
- `ISOCoupledResource$setOperationName()`
- `ISOCoupledResource$setIdentifier()`
- `ISOCoupledResource$clone()`

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

Usage:

```
ISOCoupledResource$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setOperationName()`**:** Set operation name

Usage:

```
ISOCoupledResource$setOperationName(operationName, locales = NULL)
```

Arguments:

`operationName` operation name

`locales` a list of localized names. Default is `NULL`

Method `setIdentifier()`**:** Set identifier

Usage:

```
ISOCoupledResource$setIdentifier(identifier, locales = NULL)
```

Arguments:

`identifier` identifier

`locales` a list of localized identifiers. Default is `NULL`

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

Usage:

`ISOCoupledResource$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19119:2005 - Geographic information – Services

Examples

```
md <- ISOCoupledResource$new()
md$setOperationName("name")
md$setIdentifier("identifier")
xml <- md$encode()
```

`ISOCouplingType` *ISOCouplingType*

Description

ISOCouplingType

ISOCouplingType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOCouplingType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOCouplingType`

Methods

Public methods:

- `ISOCouplingType$new()`
- `ISOCouplingType$clone()`

Method `new()`: Initializes object

Usage:

`ISOCouplingType$new(xml = NULL, value, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOCouplingType$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19119:2005 - Geographic information – Services

Examples

```
#possible values
values <- ISOCouplingType$values(labels = TRUE)

#couplingType
couplingType <- ISOCouplingType$new(value = "loose")
```

`ISOCoverageContentType`

ISOCoverageContentType

Description

ISOCoverageContentType

ISOCoverageContentType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO CoverageContentType

Methods

`new(xml,value, description)` This method is used to instantiate an `ISOCoverageContentType`

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOCoverageContentType`

Methods

Public methods:

- `ISOCoverageContentType$new()`
- `ISOCoverageContentType$clone()`

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

Usage:

```
ISOCoverageContentType$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOCoverageContentType$values(labels = TRUE)

#example of CoverageContentType
modelResultType <- ISOCoverageContentType$new(value = "modelResult")
```

`ISOCoverageDescription` *ISOCoverageDescription*

Description

ISOCoverageDescription

ISOCoverageDescription

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOCoverageDescription

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOContentInformation` -> `ISOCoverageDescription`

Public fields

`attributeDescription` attributeDescription: ISoRecordType

`contentType` contentType: ISOCoverageContentType

`dimension` dimension: ISORangeDimension

Methods

Public methods:

- `ISOCoverageDescription$new()`
- `ISOCoverageDescription$setAttributeDescription()`
- `ISOCoverageDescription$setContentType()`
- `ISOCoverageDescription$addDimension()`
- `ISOCoverageDescription$delDimension()`
- `ISOCoverageDescription$clone()`

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

Usage:

```
ISOCoverageDescription$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setAttributeDescription()`**:** Set attribute description

Usage:

```
ISOCoverageDescription$setAttributeDescription(attributeDescription)
```

Arguments:

`attributeDescription` attribute description, object of class ISORecordType or character

Method `setContentType()`**:** Set content type

Usage:

```
ISOCoverageDescription$setContentType(contentType)
```

Arguments:

`contentType` contentType, object of class ISOCoverageContentType or character

Method `addDimension()`**:** Adds dimension

Usage:

```
ISOCoverageDescription$addDimension(dimension)
```

Arguments:

`dimension` object of class ISORangeDimension

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDimension()`**:** Deletes dimension

Usage:

```
ISOCoverageDescription$delDimension(dimension)
```

Arguments:

`dimension` object of class ISORangeDimension

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create coverage description
md <- ISOCoverageDescription$new()
md$setAttributeDescription("test")
md$setContentType("modelResult")

#adding 3 arbitrary dimensions
for(i in 1:3){
   band <- ISOBand$new()
 mn <- ISOMemberName$new(aName = sprintf("name %s",i), attributeType = sprintf("type %s",i))
   band$setSequenceIdentifier(mn)
   band$setDescriptor("descriptor")
   band$setMaxValue(10)
   band$setMinValue(1)
   gml <- GMLBaseUnit$new(id = sprintf("ID%s",i))
   gml$setDescriptionReference("someref")
   gml$setIdentifier("identifier", "codespace")
   gml$addName("name1", "codespace")
   gml$addName("name2", "codespace")
   gml$setQuantityTypeReference("someref")
   gml$setCatalogSymbol("symbol")
   gml$setUnitsSystem("somelink")
   band$setUnits(gml)
   band$setPeakResponse(9)
   band$setBitsPerValue(5)
   band$setToneGradation(100)
   band$setScaleFactor(1)
   band$setOffset(4)
   md$addDimension(band)
}
xml <- md$encode()
```

`ISODataFile` *ISODataFile*

Description

ISODataFile

ISODataFile

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO DataFile

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISODataFile`

Public fields

`fileName` fileName [1..1]: ISOFileName

`fileDescription` fileDescription [1..1]: character|ISOLocalisedCharacterString

`fileType` fileType [1..1]: ISOMimeFileType

`featureTypes` featureTypes [0..*]: ISOLocalName|ISOScopedName

`fileFormat` fileFormat [1..1]: ISOFormat

Methods

Public methods:

- `ISODataFile$new()`
- `ISODataFile$setFileName()`
- `ISODataFile$setFileDescription()`
- `ISODataFile$setFileType()`
- `ISODataFile$addFeatureType()`
- `ISODataFile$delFeatureType()`
- `ISODataFile$setFileFormat()`
- `ISODataFile$clone()`

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

Usage:

```
ISODataFile$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setFileName()`**:** Set file name

Usage:

```
ISODataFile$setFileName(fileName)
```

Arguments:

`fileName` object of class ISOFileName

Method `setFileDescription()`**:** Set file description

Usage:

```
ISODataFile$setFileDescription(fileDescription, locales = NULL)
```

Arguments:

`fileDescription` object of class character

`locales` list of localized descriptions. Default is `NULL`

Method `setFileType()`**:** Set file type

Usage:

`ISODataFile$setFileType(fileType)`

Arguments:

`fileType` object of class ISOMimeFileType

Method `addFeatureType()`**:** Adds feature type

Usage:

`ISODataFile$addFeatureType(featureType)`

Arguments:

`featureType` object of class ISOLocalName, ISOScopedName or character

Returns: `TRUE` if added, `FALSE` otherwise

Method `delFeatureType()`**:** Deletes feature type

Usage:

`ISODataFile$delFeatureType(featureType)`

Arguments:

`featureType` object of class ISOLocalName, ISOScopedName or character

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setFileFormat()`**:** Set file format

Usage:

`ISODataFile$setFileFormat(fileFormat)`

Arguments:

`fileFormat` file format, object of class ISOFormat

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

Usage:

`ISODataFile$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19139:2007 Geographic information – XML

Examples

```
md <- ISODataFile$new()
md$setFileName(ISOFileName$new(file = "someuri", name = "filename"))
md$setFileDescription("description")
md$setFileType(ISOMimeFileType$new(type = "somemimetype", name = "Mime type name"))
md$addFeatureType("feature_type")
f <- ISOFormat$new()
f$setName("name")
f$setVersion("1.0")
f$setAmendmentNumber("2")
f$setSpecification("specification")
md$setFileFormat(f)
xml <- md$encode()
```

`ISODataIdentification` *ISODataIdentification*

Description

ISODataIdentification

ISODataIdentification

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO DataIdentification

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOIdentification` -> `ISODataIdentification`

Public fields

`spatialRepresentationType` spatialRepresentationType [0..*]: ISOSpatialRepresentationType

`spatialResolution` spatialResolution [0..*]: ISOResolution

`language` language [1..*]: character

`characterSet` characterSet [0..*]: ISOCharacterSet

`topicCategory` topicCategory [0..*]: ISOTopicCategory

`extent` extent [0..*]: ISOExtent

`supplementalInformation` supplementalInformation

Methods

Public methods:

- `ISODataIdentification$new()`
- `ISODataIdentification$addSpatialRepresentationType()`
- `ISODataIdentification$setSpatialRepresentationType()`
- `ISODataIdentification$delSpatialRepresentationType()`
- `ISODataIdentification$addSpatialResolution()`
- `ISODataIdentification$delSpatialResolution()`
- `ISODataIdentification$addLanguage()`
- `ISODataIdentification$setLanguage()`
- `ISODataIdentification$delLanguage()`
- `ISODataIdentification$addCharacterSet()`
- `ISODataIdentification$setCharacterSet()`
- `ISODataIdentification$delCharacterSet()`
- `ISODataIdentification$addTopicCategory()`
- `ISODataIdentification$setTopicCategory()`
- `ISODataIdentification$delTopicCategory()`
- `ISODataIdentification$addExtent()`
- `ISODataIdentification$setExtent()`
- `ISODataIdentification$delExtent()`
- `ISODataIdentification$setSupplementalInformation()`
- `ISODataIdentification$clone()`

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

Usage:

```
ISODataIdentification$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addSpatialRepresentationType()`**:** Adds spatial representation type

Usage:

```
ISODataIdentification$addSpatialRepresentationType(spatialRepresentationType)
```

Arguments:

`spatialRepresentationType` object of class ISOSpatialRepresentationType or any character among values returned by `ISOSpatialRepresentationType$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `setSpatialRepresentationType()`**:** Sets spatial representation type

Usage:

```
ISODataIdentification$setSpatialRepresentationType(spatialRepresentationType)
```

Arguments:

`spatialRepresentationType` object of class ISOSpatialRepresentationType or any character among values returned by `ISOSpatialRepresentationType$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSpatialRepresentationType()`**:** Deletes spatial representation type

Usage:

```
ISODataIdentification$delSpatialRepresentationType(spatialRepresentationType)
```

Arguments:

`spatialRepresentationType` object of class ISOSpatialRepresentationType or any character among values returned by `ISOSpatialRepresentationType$values()`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addSpatialResolution()`**:** Adds spatial resolution

Usage:

```
ISODataIdentification$addSpatialResolution(resolution)
```

Arguments:

`resolution` object of class ISOResolution

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSpatialResolution()`**:** Deletes spatial resolution

Usage:

```
ISODataIdentification$delSpatialResolution(resolution)
```

Arguments:

`resolution` object of class ISOResolution

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addLanguage()`**:** Adds language

Usage:

```
ISODataIdentification$addLanguage(locale)
```

Arguments:

`locale` object of class ISOLanguage or any character value among those returned by `ISOLanguage$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `setLanguage()`**:** Sets language

Usage:

```
ISODataIdentification$setLanguage(locale)
```

Arguments:

`locale` object of class ISOLanguage or any character value among those returned by `ISOLanguage$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delLanguage()`**:** Deletes language

Usage:

`ISODataIdentification$delLanguage(locale)`

Arguments:

`locale` object of class ISOLanguage or any character value among those returned by `ISOLanguage$values()`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addCharacterSet()`**:** Adds character set

Usage:

`ISODataIdentification$addCharacterSet(charset)`

Arguments:

`charset` object of class ISOCharacterSet or any character value among those returned by `ISOCharacterSet$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `setCharacterSet()`**:** Sets character set

Usage:

`ISODataIdentification$setCharacterSet(charset)`

Arguments:

`charset` object of class ISOCharacterSet or any character value among those returned by `ISOCharacterSet$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delCharacterSet()`**:** Deletes character set

Usage:

`ISODataIdentification$delCharacterSet(charset)`

Arguments:

`charset` object of class ISOCharacterSet or any character value among those returned by `ISOCharacterSet$values()`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addTopicCategory()`**:** Adds topic category

Usage:

`ISODataIdentification$addTopicCategory(topicCategory)`

Arguments:

`topicCategory` object of class ISOTopicCategory or any character value among those returned by `ISOTopicCategory$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `setTopicCategory()`**:** Sets topic category

Usage:

`ISODataIdentification$setTopicCategory(topicCategory)`

Arguments:

`topicCategory` object of class ISOTopicCategory or any character value topicCategory those returned by `ISOTopicCategory$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delTopicCategory()`**:** Deletes topic category

Usage:

```
ISODataIdentification$delTopicCategory(topicCategory)
```

Arguments:

`topicCategory` object of class ISOTopicCategory or any character value among those returned by `ISOTopicCategory$values()`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addExtent()`**:** Adds extent

Usage:

```
ISODataIdentification$addExtent(extent)
```

Arguments:

`extent` object of class ISOExtent

Returns: `TRUE` if added, `FALSE` otherwise

Method `setExtent()`**:** Sets extent

Usage:

```
ISODataIdentification$setExtent(extent)
```

Arguments:

`extent` object of class ISOExtent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delExtent()`**:** Deletes extent

Usage:

```
ISODataIdentification$delExtent(extent)
```

Arguments:

`extent` object of class ISOExtent

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setSupplementalInformation()`**:** Set supplemental information

Usage:

```
ISODataIdentification$setSupplementalInformation(
  supplementalInformation,
  locales = NULL
)
```

Arguments:

`supplementalInformation` supplemental information

`locales` a list of localized information. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create dataIdentification
md <- ISODataIdentification$new()
md$setAbstract("abstract")
md$setPurpose("purpose")
md$addLanguage("eng")
md$addCharacterSet("utf8")
md$addTopicCategory("biota")
md$addTopicCategory("oceans")

#adding a point of contact
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
rp$setOrganisationName("somewhere")
rp$setPositionName("someposition")
rp$setRole("pointOfContact")
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact$setOnlineResource(res)
rp$setContactInfo(contact)
md$addPointOfContact(rp)

#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(ISOdate(2015, 1, 1, 1))
```

```
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp)
md$setCitation(ct)

#graphic overview
go <- ISOBrowseGraphic$new(
  fileName = "http://wwww.somefile.org/png",
  fileDescription = "Map Overview",
  fileType = "image/png"
)
md$addGraphicOverview(go)

#maintenance information
mi <- ISOMaintenanceInformation$new()
mi$setMaintenanceFrequency("daily")
md$addResourceMaintenance(mi)

#adding legal constraints
lc <- ISOLegalConstraints$new()
lc$addUseLimitation("limitation1")
lc$addUseLimitation("limitation2")
lc$addUseLimitation("limitation3")
lc$addAccessConstraint("copyright")
lc$addAccessConstraint("license")
lc$addUseConstraint("copyright")
lc$addUseConstraint("license")
md$addResourceConstraints(lc)

#adding extent
extent <- ISOExtent$new()
bbox <- ISOGeographicBoundingBox$new(minx = -180, miny = -90, maxx = 180, maxy = 90)
extent$addGeographicElement(bbox)
md$addExtent(extent)

#add keywords
kwds <- ISOKeywords$new()
kwds$addKeyword("keyword1")
kwds$addKeyword("keyword2")
kwds$setKeywordType("theme")
th <- ISOCitation$new()
th$setTitle("General")
th$addDate(d)
kwds$setThesaurusName(th)
md$addKeywords(kwds)

#supplementalInformation
md$setSupplementalInformation("some additional information")

xml <- md$encode()
```

`ISODataQuality` *ISODataQuality*

Description

ISODataQuality

ISODataQuality

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO DataQuality

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISODataQuality`

Public fields

`scope` scope

`report` list of reports

`lineage` lineage

Methods

Public methods:

- `ISODataQuality$new()`
- `ISODataQuality$setScope()`
- `ISODataQuality$addReport()`
- `ISODataQuality$setLineage()`
- `ISODataQuality$clone()`

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

Usage:

```
ISODataQuality$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setScope()`**:** Set scope

Usage:

```
ISODataQuality$setScope(scope)
```

Arguments:

`scope` scope

Method `addReport()`: Adds report

Usage:

`ISODataQuality$addReport(report)`

Arguments:

`report` report, object of class ISODomainConsistency

Returns: `TRUE` if added, `FALSE` otherwise

Method `setLineage()`: Set lineage

Usage:

`ISODataQuality$setLineage(lineage)`

Arguments:

`lineage` lineage, object of class ISOLineage

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

Usage:

`ISODataQuality$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create dataQuality object with a 'dataset' scope
dq <- ISODataQuality$new()
scope <- ISOScope$new()
scope$setLevel("dataset")
dq$setScope(scope)

#add data quality reports...

#add a report the data quality
dc <- ISODomainConsistency$new()
result <- ISOConformanceResult$new()
spec <- ISOCitation$new()
spec$setTitle("Data Quality check")
spec$addAlternateTitle("This is is some data quality check report")
d <- ISODate$new()
```

```
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dc$addResult(result)
dq$addReport(dc)

#add INSPIRE reports?
#INSPIRE - interoperability of spatial data sets and services
dc_inspire1 <- ISODomainConsistency$new()
cr_inspire1 <- ISOConformanceResult$new()
cr_inspire_spec1 <- ISOCitation$new()
cr_title <- paste(
 "Commission Regulation (EU) No 1089/2010 of 23 November 2010 implementing Directive 2007/2/EC",
 "of the European Parliament and of the Council as regards interoperability of spatial data",
  "sets and services"
)
cr_inspire_spec1$setTitle(cr_title)
cr_inspire1$setExplanation("See the referenced specification")
cr_inspire_date1 <- ISODate$new()
cr_inspire_date1$setDate(ISOdate(2010,12,8))
cr_inspire_date1$setDateType("publication")
cr_inspire_spec1$addDate(cr_inspire_date1)
cr_inspire1$setSpecification(cr_inspire_spec1)
cr_inspire1$setPass(TRUE)
dc_inspire1$addResult(cr_inspire1)
dq$addReport(dc_inspire1)
#INSPIRE - metadata
dc_inspire2 <- ISODomainConsistency$new()
cr_inspire2 <- ISOConformanceResult$new()
cr_inspire_spec2 <- ISOCitation$new()
cr_title2 <- paste(
 "COMMISSION REGULATION (EC) No 1205/2008 of 3 December 2008 implementing Directive 2007/2/EC",
  "of the European Parliament and of the Council as regards metadata"
)
cr_inspire_spec2$setTitle(cr_title2)
cr_inspire2$setExplanation("See the referenced specification")
cr_inspire_date2 <- ISODate$new()
cr_inspire_date2$setDate(ISOdate(2008,12,4))
cr_inspire_date2$setDateType("publication")
cr_inspire_spec2$addDate(cr_inspire_date2)
cr_inspire2$setSpecification(cr_inspire_spec2)
cr_inspire2$setPass(TRUE)
dc_inspire2$addResult(cr_inspire2)
dq$addReport(dc_inspire2)

#add lineage (more example of lineages in ISOLineage documentation)
lineage <- ISOLineage$new()
lineage$setStatement("statement")
dq$setLineage(lineage)
```

```
#xml
xml <- dq$encode()
```

ISODataQualityAbstractElement
ISODataQualityAbstractElement

Description

ISODataQualityAbstractElement

ISODataQualityAbstractElement

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISODataQualityAbstractElement

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISODataQualityAbstractElement`

Public fields

`nameOfMeasure` nameOfMeasure [0..*]: character

`measureIdentification` measureIdentification [0..1]: ISOMetaIdentifier

`measureDescription` measureDescription [0..1]: character

`evaluationMethodType` evaluationMethodType [0..1]: ISOEvaluationMethodType

`evaluationMethodDescription` evaluationMethodDescription [0..1]: character

`evaluationProcedure` evaluationProcedure [0..1]: ISOCitation

`dateTime` dateTime [0..1]: ISODateTime

`result` result [1..2]: ISOConformanceResult

Methods

Public methods:

- `ISODataQualityAbstractElement$new()`
- `ISODataQualityAbstractElement$addNameOfMeasure()`
- `ISODataQualityAbstractElement$delNameOfMeasure()`
- `ISODataQualityAbstractElement$setMeasureIdentification()`
- `ISODataQualityAbstractElement$setMeasureDescription()`

- `ISODataQualityAbstractElement$setEvaluationMethodType()`
- `ISODataQualityAbstractElement$setEvaluationMethodDescription()`
- `ISODataQualityAbstractElement$setEvaluationProcedure()`
- `ISODataQualityAbstractElement$setDateTime()`
- `ISODataQualityAbstractElement$addResult()`
- `ISODataQualityAbstractElement$delResult()`
- `ISODataQualityAbstractElement$clone()`

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

Usage:

```
ISODataQualityAbstractElement$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addNameOfMeasure()`**:** Adds name of measure

Usage:

```
ISODataQualityAbstractElement$addNameOfMeasure(name, locales = NULL)
```

Arguments:

`name` name

`locales` list of localized names. Default is `NULL`

Returns: `TRUE` if added, `FALSE`

Method `delNameOfMeasure()`**:** Deletes name of measure

Usage:

```
ISODataQualityAbstractElement$delNameOfMeasure(name, locales = NULL)
```

Arguments:

`name` name

`locales` list of localized names. Default is `NULL`

Returns: `TRUE` if deleted, `FALSE`

Method `setMeasureIdentification()`**:** Set measure identification

Usage:

```
ISODataQualityAbstractElement$setMeasureIdentification(identification)
```

Arguments:

`identification` object of class ISOMetaIdentifier

Method `setMeasureDescription()`**:** Set measure description

Usage:

```
ISODataQualityAbstractElement$setMeasureDescription(
  description,
  locales = NULL
)
```

Arguments:

`description` object of class character

`locales` list of localized descriptions. Default is `NULL`

Method `setEvaluationMethodType():` Set evaluation method type

Usage:

```
ISODataQualityAbstractElement$setEvaluationMethodType(type)
```

Arguments:

`type` object of class ISOEvaluationMethodType or any character value from those returned by `ISOEvaluationMethodType$values()`

Method `setEvaluationMethodDescription():` Set evaluation method description

Usage:

```
ISODataQualityAbstractElement$setEvaluationMethodDescription(
  description,
  locales = NULL
)
```

Arguments:

`description` description

`locales` list of localized descriptions. Default is `NULL`

Method `setEvaluationProcedure():` Set evaluation procedure

Usage:

```
ISODataQualityAbstractElement$setEvaluationProcedure(procedure)
```

Arguments:

`procedure` procedure, object of class ISOCitation

Method `setDateTime():` Set date time

Usage:

```
ISODataQualityAbstractElement$setDateTime(dateTime)
```

Arguments:

`dateTime` date time, object of class POSIXct

Method `addResult():` Adds result

Usage:

```
ISODataQualityAbstractElement$addResult(result)
```

Arguments:

`result` object of class ISOConformanceResult

Returns: `TRUE` if added, `FALSE` otherwise

Method `delResult():` Deletes result

Usage:

```
ISODataQualityAbstractElement$delResult(result)
```

Arguments:

`result` object of class ISOConformanceResult

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISODataSet` *ISODataSet*

Description

ISODataSet

ISODataSet

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISODataSet

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISODataSet`

Public fields

`has` has [1..*]

`partOf` partOf [0..*]

Methods

Public methods:

- `ISODataSet$new()`
- `ISODataSet$addHasMetadata()`
- `ISODataSet$delHasMetadata()`
- `ISODataSet$addPartOf()`
- `ISODataSet$delPartOf()`
- `ISODataSet$clone()`

Method `new()`: Initializes object

Usage:

```
ISODataSet$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addHasMetadata()`: Adds metadata

Usage:

```
ISODataSet$addHasMetadata(metadata)
```

Arguments:

`metadata` metadata, object of class ISOMetadata

Returns: `TRUE` if added, `FALSE` otherwise

Method `delHasMetadata()`: Deletes metadata

Usage:

```
ISODataSet$delHasMetadata(metadata)
```

Arguments:

`metadata` metadata, object of class ISOMetadata

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addPartOf()`: Adds aggregate dataset is part of

Usage:

```
ISODataSet$addPartOf(partOf)
```

Arguments:

`partOf` object inheriting class ISOAbstractAggregate

Returns: `TRUE` if added, `FALSE` otherwise

Method `delPartOf()`: Deletes aggregate dataset is part of

Usage:

```
ISODataSet$delPartOf(partOf)
```

Arguments:

`partOf` object inheriting class ISOAbstractAggregate

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISODatatype` *ISODatatype*

Description

ISODatatype

ISODatatype

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Datatype

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISODatatype`

Methods

Public methods:

- `ISODatatype$new()`
- `ISODatatype$clone()`

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

Usage:

```
ISODatatype$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISODatatype$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISODatatype$values(labels = TRUE)

#string Datatype
stringType <- ISODatatype$new(value = "characterString")
```

`ISODate` *ISODate*

Description

ISODate

ISODate

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Date

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISODate`

Public fields

`date` date

`dateType` date type

Methods

Public methods:

- `ISODate$new()`
- `ISODate$setDate()`
- `ISODate$setDateType()`
- `ISODate$clone()`

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

Usage:

```
ISODate$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setDate()`**:** Set date

Usage:

```
ISODate$setDate(date)
```

Arguments:

`date` object of class Date or POSIXct

Method `setDateType()`**:** Set date type

Usage:

```
ISODate$setDateType(dateType)
```

Arguments:

`dateType` object of class ISODateType or any character values returned by `ISODateType$values()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISODate$new()
md$setDate(ISOdate(2015, 1, 1, 1))
md$setDateType("publication")
xml <- md$encode()
```

`ISODateType` *ISODateType*

Description

ISODateType

ISODateType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO DateType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISODateType`

Methods

Public methods:

- `ISODateType$new()`
- `ISODateType$clone()`

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

Usage:

`ISODateType$new(xml = NULL, value = NULL, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISODateType$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISODateType$values(labels = TRUE)

#creation datetype
creation <- ISODateType$new(value = "creation")
```

ISODCPList *ISODCPList*

Description

ISODCPList

ISODCPList

Format

R6Class object.

Value

Object of R6Class for modelling an ISO DCPList

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISOCodeListValue -> ISODCPList

Methods

Public methods:

- ISODCPList$new()
- ISODCPList$clone()

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

Usage:

```
ISODCPList$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISODCPList$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19119:2005 - Geographic information – Service

Examples

```
#possible values
values <- ISODCPList$values(labels = TRUE)

#example
javaDCP <- ISODCPList$new(value = "JAVA")
```

`ISODefinitionReference` *ISODefinitionReference*

Description

ISODefinitionReference

ISODefinitionReference

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISODefinitionReference

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISODefinitionReference`

Public fields

`sourceIdentifier` sourceIdentifier [0..1]: character

`definitionSource` definitionSource: ISODefinitionSource

Methods

Public methods:

- `ISODefinitionReference$new()`
- `ISODefinitionReference$setSourceIdentifier()`
- `ISODefinitionReference$setDefinitionSource()`
- `ISODefinitionReference$clone()`

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

Usage:

```
ISODefinitionReference$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setSourceIdentifier()`**:** Set source identifier

Usage:

```
ISODefinitionReference$setSourceIdentifier(identifier)
```

Arguments:

`identifier` identifier

Method `setDefinitionSource()`**:** Set definition source

Usage:

```
ISODefinitionReference$setDefinitionSource(source)
```

Arguments:

`source` object of class ISODefinitionSource or ISOCitation

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

ISODefinitionSource *ISODefinitionSource*

Description

ISODefinitionSource

ISODefinitionSource

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISODefinitionSource

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISODefinitionSource`

Public fields

`source` source [0..1]: ISOCitation

Methods

Public methods:

- `ISODefinitionSource$new()`
- `ISODefinitionSource$setSource()`
- `ISODefinitionSource$clone()`

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

Usage:

```
ISODefinitionSource$new(xml = NULL, source = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`source` source object of class ISOCitation

Method `setSource()`**:** Set source

Usage:

```
ISODefinitionSource$setSource(source)
```

Arguments:

`source` object of class ISOCitation

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

`ISODigitalTransferOptions` *ISODigitalTransferOptions*

Description

ISODigitalTransferOptions

ISODigitalTransferOptions

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO DigitalTransferOptions

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISODigitalTransferOptions`

Public fields

`unitsOfDistribution` unitsOfDistribution [0..1]: character

`transferSize` transferSize [0..1]: integer

`onLine` onLine [0..*]: ISOOnlineResource

`offLine` offLine [0..1]: MD_Medium

Methods

Public methods:

- `ISODigitalTransferOptions$new()`
- `ISODigitalTransferOptions$setUnitsOfDistribution()`
- `ISODigitalTransferOptions$setTransferSize()`
- `ISODigitalTransferOptions$addOnlineResource()`
- `ISODigitalTransferOptions$setOnlineResource()`
- `ISODigitalTransferOptions$delOnlineResource()`
- `ISODigitalTransferOptions$addOfflineResource()`
- `ISODigitalTransferOptions$setOfflineResource()`
- `ISODigitalTransferOptions$delOfflineResource()`
- `ISODigitalTransferOptions$clone()`

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

Usage:

```
ISODigitalTransferOptions$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setUnitsOfDistribution()`**:** Set units of distribution

Usage:

```
ISODigitalTransferOptions$setUnitsOfDistribution(unit)
```

Arguments:

`unit` unit

Method `setTransferSize()`**:** Set transfer size

Usage:

```
ISODigitalTransferOptions$setTransferSize(transferSize)
```

Arguments:

`transferSize` transfer size

Method `addOnlineResource()`**:** Adds online resource

Usage:

```
ISODigitalTransferOptions$addOnlineResource(onlineResource)
```

Arguments:

`onlineResource` object of class ISOOnlineResource

Returns: `TRUE` if added, `FALSE` otherwise

Method `setOnlineResource()`**:** Sets online resource

Usage:

```
ISODigitalTransferOptions$setOnlineResource(onlineResource)
```

Arguments:

`onlineResource` object of class ISOOnlineResource

Returns: `TRUE` if added, `FALSE` otherwise

Method `delOnlineResource()`**:** Deletes online resource

Usage:

```
ISODigitalTransferOptions$delOnlineResource(onlineResource)
```

Arguments:

`onlineResource` object of class ISOOnlineResource

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addOfflineResource()`**:** Adds offline resource

Usage:

```
ISODigitalTransferOptions$addOfflineResource(offlineResource)
```

Arguments:

`offlineResource` object of class ISOMedium

Returns: `TRUE` if added, `FALSE` otherwise

Method `setOfflineResource()`**:** Sets offline resource

Usage:

```
ISODigitalTransferOptions$setOfflineResource(offlineResource)
```

Arguments:

`offlineResource` object of class ISOMedium

Returns: `TRUE` if added, `FALSE` otherwise

Method `delOfflineResource()`**:** Deletes offline resource

Usage:

```
ISODigitalTransferOptions$delOfflineResource(offlineResource)
```

Arguments:

`offlineResource` object of class ISOMedium

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISODigitalTransferOptions$new()

or <- ISOOnlineResource$new()
or$setLinkage("http://somelink")
or$setName("name")
or$setDescription("description")
or$setProtocol("WWW:LINK-1.0-http--link")
md$addOnlineResource(or)

xml <- md$encode()
```

`ISODimension` *ISODimension*

Description

ISODimension

ISODimension

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Dimension

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISODimension`

Public fields

`dimensionName` dimensionName [1..1]: ISODimensionNameType

`dimensionSize` dimensionSize [1..1]: integer

`resolution` resolution [0..1]: ISOMeasure or subclass

Methods

Public methods:

- `ISODimension$new()`
- `ISODimension$setName()`
- `ISODimension$setSize()`
- `ISODimension$setResolution()`

- ISODimension$clone()

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

Usage:

```
ISODimension$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setName()`**:** Set name

Usage:

```
ISODimension$setName(name)
```

Arguments:

`name` object of class ISODimensionNameType or any character among values returned by `ISODimensionNameType$valu`

Method `setSize()`**:** Set size

Usage:

```
ISODimension$setSize(size)
```

Arguments:

`size` object of class integer

Method `setResolution()`**:** Sets the resolution

Usage:

```
ISODimension$setResolution(resolution)
```

Arguments:

`resolution` object of class `ISOMeasure` or any subclass `ISOLength`, `ISODistance`, `ISOAngle`, `ISOScale`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create dimension
md <- ISODimension$new()
md$setName("row")
md$setSize(1)
md$setResolution(ISOLength$new(value=1,uom="m"))
xml <- md$encode()
```

`ISODimensionNameType` *ISODimensionNameType*

Description

ISODimensionNameType

ISODimensionNameType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO DimensionNameType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISODimensionNameType`

Methods

Public methods:

- `ISODimensionNameType$new()`
- `ISODimensionNameType$clone()`

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

Usage:

```
ISODimensionNameType$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISODimensionNameType$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISODimensionNameType$values(labels = TRUE)

#row DimensionNameType
rowType <- ISODimensionNameType$new(value = "row")
```

`ISODistance` *ISODistance*

Description

ISODistance

ISODistance

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Distance measure

Methods

`new(xml,value, uom, useUomURI)` This method is used to instantiate an ISODistance. The `uom` argument represents the symbol of unit of measure used. The parameter `useUomURI` can be used to set the uom as URI, its default value is `FALSE`.

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOMeasure` -> `geometa::ISOLength` -> `ISODistance`

Methods

Public methods:

- `ISODistance$new()`
- `ISODistance$clone()`

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

Usage:

```
ISODistance$new(xml = NULL, value, uom, useUomURI = FALSE)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`uom` uom symbol of unit of measure used

`useUomURI` use uom URI. Default is `FALSE`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISODistribution` *ISODistribution*

Description

ISODistribution

ISODistribution

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Distribution

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISODistribution`

Public fields

`distributionFormat` distributionFormat [0..*]: ISOFormat

`distributor` distributor [0..*]: ISODistributor

`transferOptions` transferOptions [0..*]: ISODigitalTransferOptions

Methods

Public methods:

- `ISODistribution$new()`
- `ISODistribution$addFormat()`
- `ISODistribution$delFormat()`
- `ISODistribution$addDistributor()`
- `ISODistribution$delDistributor()`
- `ISODistribution$addDigitalTransferOptions()`
- `ISODistribution$setDigitalTransferOptions()`
- `ISODistribution$delDigitalTransferOptions()`
- `ISODistribution$clone()`

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

Usage:

```
ISODistribution$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addFormat()`**:** Adds format

Usage:

```
ISODistribution$addFormat(format)
```

Arguments:

`format` format object of class ISOFormat

Returns: `TRUE` if added, `FALSE` otherwise

Method `delFormat()`**:** Deletes format

Usage:

```
ISODistribution$delFormat(format)
```

Arguments:

`format` format object of class ISOFormat

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addDistributor()`**:** Adds distributor

Usage:

```
ISODistribution$addDistributor(distributor)
```

Arguments:

`distributor` distributor object of class ISODistributor

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDistributor()`**:** Deletes distributor

Usage:

```
ISODistribution$delDistributor(distributor)
```

Arguments:

`distributor` distributor object of class ISODistributor

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addDigitalTransferOptions()`**:** Adds digital transfer options

Usage:

```
ISODistribution$addDigitalTransferOptions(options)
```

Arguments:

`options` options object of class ISODigitalTransferOptions

Returns: `TRUE` if added, `FALSE` otherwise

Method `setDigitalTransferOptions()`**:** Sets digital transfer options

Usage:

```
ISODistribution$setDigitalTransferOptions(options)
```

Arguments:

`options` options object of class ISODigitalTransferOptions

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDigitalTransferOptions()`**:** Deletes digital transfer options

Usage:

```
ISODistribution$delDigitalTransferOptions(options)
```

Arguments:

`options` options object of class ISODigitalTransferOptions

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISODistribution$new()

dto <- ISODigitalTransferOptions$new()
for(i in 1:3){
 or <- ISOOnlineResource$new()
 or$setLinkage(paste0("http://somelink",i))
 or$setName(paste0("name",i))
 or$setDescription(paste0("description",i))
 or$setProtocol("WWW:LINK-1.0-http--link")
 dto$addOnlineResource(or)
}
md$setDigitalTransferOptions(dto)

xml <- md$encode()
```

`ISODistributionUnits` *ISODistributionUnits*

Description

ISODistributionUnits

ISODistributionUnits

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO DistributionUnits

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISODistributionUnits`

Methods

Public methods:

- `ISODistributionUnits$new()`
- `ISODistributionUnits$clone()`

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

Usage:

```
ISODistributionUnits$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
unit <- ISODistributionUnits$new(value = "unit")
```

`ISODistributor` *ISODistributor*

Description

ISODistributor

ISODistributor

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISODistributor

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISODistributor`

Public fields

`distributorContact` distributorContact : ISOResponsibleParty

`distributorFormat` distributorFormat : ISOFormat

Methods

Public methods:

- `ISODistributor$new()`
- `ISODistributor$setContact()`
- `ISODistributor$addFormat()`
- `ISODistributor$delFormat()`
- `ISODistributor$clone()`

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

Usage:

```
ISODistributor$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setContact()`**:** Set contact

Usage:

```
ISODistributor$setContact(contact)
```

Arguments:

`contact` object of class ISOResponsibleParty

Method `addFormat()`**:** Adds format

Usage:

```
ISODistributor$addFormat(format)
```

Arguments:

`format` format object of class ISOFormat

Returns: `TRUE` if added, `FALSE` otherwise

Method `delFormat()`**:** Deletes format

Usage:

```
ISODistributor$delFormat(format)
```

Arguments:

`format` format object of class ISOFormat

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISODistributor$new()
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
rp$setOrganisationName("somewhere")
rp$setPositionName("Data manager")

contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact$setOnlineResource(res)
rp$setContactInfo(contact)
rp$setRole("author")
md$setContact(rp)

format <- ISOFormat$new()
format$setName("name")
format$setVersion("1.0")
format$setAmendmentNumber("2")
format$setSpecification("specification")
md$addFormat(format)

xml <- md$encode()
```

ISODomainConsistency *ISODomainConsistency*

Description

ISODomainConsistency

ISODomainConsistency

Format

R6Class object.

Value

Object of R6Class for modelling an ISODomainConsistency

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISODataQualityAbstractElement -> geometa::ISOAbstractLogicalConsistency -> ISODomainConsistency

Methods

Public methods:

- ISODomainConsistency$clone()

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

Usage:

`ISODomainConsistency$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISODomainConsistency$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
```

```
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

ISOElementSequence *ISOElementSequence*

Description

ISOElementSequence

ISOElementSequence

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOElementSequence

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOElementSequence`

Methods

Public methods:

- `ISOElementSequence$new()`
- `ISOElementSequence$clone()`

Method `new()`**:** Initializes sequence object

Usage:

`ISOElementSequence$new(xml = NULL, ...)`

Arguments:

`xml` object of class XMLInternalNode-class

`...` other args

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

This class is used internally by geometa to deal with simple type not handled by proper class element. e.g. `name` property of `ISOParameter` class from ISO 19119:2005

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOEvaluationMethodType`

ISOEvaluationMethodType

Description

ISOEvaluationMethodType

ISOEvaluationMethodType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO EvaluationMethodType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOEvaluationMethodType`

Methods

Public methods:

- `ISOEvaluationMethodType$new()`
- `ISOEvaluationMethodType$clone()`

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

Usage:

`ISOEvaluationMethodType$new(xml = NULL, value, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOEvaluationMethodType$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOEvaluationMethodType$values(labels = TRUE)

#example of EvaluationMethodType
indirect <- ISOEvaluationMethodType$new(value = "indirect")
```

ISOExtendedElementInformation *ISOExtendedElementInformation*

Description

ISOExtendedElementInformation

ISOExtendedElementInformation

Format

R6Class object.

Value

Object of R6Class for modelling an ISO ExtendedElementInformation

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> ISOExtendedElementInformation

Public fields

`name` name [1..1]: character

`shortName` shortName [0..1]: character

`domainCode` domainCode [0..1]: integer

`definition` definition [1..1]: character

`obligation` obligation [0..1]: ISOObligation

`condition` condition [0..1]: character

`dataType` dataType [1..1]: ISODatatype

`maximumOccurrence` maximumOccurrence [0..1]: character

`domainValue` domainValue [0..1]: character

`parentEntity` parentEntity [1..*]: character

`rule` rule [1..1]: character

`rationale` rationale [0..*]: character

`source` source [1..*]: ISOResponsibleParty

Methods

Public methods:

- `ISOExtendedElementInformation$new()`
- `ISOExtendedElementInformation$setName()`
- `ISOExtendedElementInformation$setShortName()`
- `ISOExtendedElementInformation$setDomainCode()`
- `ISOExtendedElementInformation$setDefinition()`
- `ISOExtendedElementInformation$setObligation()`
- `ISOExtendedElementInformation$setCondition()`
- `ISOExtendedElementInformation$setDatatype()`
- `ISOExtendedElementInformation$setMaximumOccurrence()`
- `ISOExtendedElementInformation$setDomainValue()`
- `ISOExtendedElementInformation$addParentEntity()`
- `ISOExtendedElementInformation$delParentEntity()`
- `ISOExtendedElementInformation$setRule()`

- `ISOExtendedElementInformation$addRationale()`
- `ISOExtendedElementInformation$delRationale()`
- `ISOExtendedElementInformation$addSource()`
- `ISOExtendedElementInformation$delSource()`
- `ISOExtendedElementInformation$clone()`

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

Usage:

```
ISOExtendedElementInformation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setName()`**:** Set name

Usage:

```
ISOExtendedElementInformation$setName(name, locales = NULL)
```

Arguments:

`name` name

`locales` list of localized names. Default is `NULL`

Method `setShortName()`**:** Set short name

Usage:

```
ISOExtendedElementInformation$setShortName(shortName, locales = NULL)
```

Arguments:

`shortName` short name

`locales` list of localized short names. Default is `NULL`

Method `setDomainCode()`**:** Set domain code

Usage:

```
ISOExtendedElementInformation$setDomainCode(domainCode)
```

Arguments:

`domainCode` domain code, object of class integer

Method `setDefinition()`**:** Set definition

Usage:

```
ISOExtendedElementInformation$setDefinition(definition, locales = NULL)
```

Arguments:

`definition` definition

`locales` list of localized definitions. Default is `NULL`

Method `setObligation()`**:** Set obligation

Usage:

```
ISOExtendedElementInformation$setObligation(obligation)
```

Arguments:

`obligation` obligation, object of class ISOObligation or any character value among those returned by `ISOObligation$values()`

Method `setCondition()`: Set condition

Usage:

```
ISOExtendedElementInformation$setCondition(condition, locales = NULL)
```

Arguments:

`condition` condition

`locales` list of localized conditions. Default is `NULL`

Method `setDatatype()`: Set data type

Usage:

```
ISOExtendedElementInformation$setDatatype(dataType)
```

Arguments:

`dataType` data type, object of class ISODatatype or any character value among those returned by `ISODatatype$values()`

Method `setMaximumOccurrence()`: Set maximum occurrence

Usage:

```
ISOExtendedElementInformation$setMaximumOccurrence(maximumOccurrence)
```

Arguments:

`maximumOccurrence` max occurrence

Method `setDomainValue()`: Set domain value

Usage:

```
ISOExtendedElementInformation$setDomainValue(domainValue)
```

Arguments:

`domainValue` domain value

Method `addParentEntity()`: Adds parent entity

Usage:

```
ISOExtendedElementInformation$addParentEntity(entity)
```

Arguments:

`entity` parent entity

Returns: `TRUE` if added, `FALSE` otherwise

Method `delParentEntity()`: Deletes parent entity

Usage:

```
ISOExtendedElementInformation$delParentEntity(entity)
```

Arguments:

`entity` parent entity

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setRule()`: Set rule

Usage:

```
ISOExtendedElementInformation$setRule(rule, locales = NULL)
```

Arguments:

`rule` rule

`locales` list of localized rules. Default is `NULL`

Method `addRationale()`: Adds rationale

Usage:

```
ISOExtendedElementInformation$addRationale(rationale, locales = NULL)
```

Arguments:

`rationale` rationale

`locales` list of localized rationales. Default is `NULL`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delRationale()`: Deletes rationale

Usage:

```
ISOExtendedElementInformation$delRationale(rationale, locales = NULL)
```

Arguments:

`rationale` rationale

`locales` list of localized rationales. Default is `NULL`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addSource()`: Adds source

Usage:

```
ISOExtendedElementInformation$addSource(source)
```

Arguments:

`source` source, object of class ISOResponsibleParty

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSource()`: Deletes source

Usage:

```
ISOExtendedElementInformation$delSource(source)
```

Arguments:

`source` source, object of class ISOResponsibleParty

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOExtendedElementInformation$new()
md$setName("name")
md$setShortName("shortName")
md$setDomainCode(1L)
md$setDefinition("some definition")
md$setObligation("mandatory")
md$setCondition("no condition")
md$setDatatype("characterString")
md$setMaximumOccurrence("string")
md$setDomainValue("value")
md$addParentEntity("none")
md$setRule("rule")
md$addRationale("rationale")

#adding a source
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
rp$setOrganisationName("somewhere")
rp$setPositionName("someposition")
rp$setRole("pointOfContact")
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact$setOnlineResource(res)
rp$setContactInfo(contact)

md$addSource(rp)

xml <- md$encode()
```

ISOExtent *ISOExtent*

Description

ISOExtent

ISOExtent

Format

R6Class object.

Value

Object of R6Class for modelling an ISO Extent

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> ISOExtent

Public fields

geographicElement	geographicElement [0..*]: ISOGeographicExtent
temporalElement	temporalElement [0..*]: ISOTemporalExtent
verticalElement	verticalElement [0..*]: ISOVerticalElement

Methods

Public methods:

- ISOExtent$new()
- ISOExtent$addGeographicElement()
- ISOExtent$setGeographicElement()
- ISOExtent$delGeographicElement()
- ISOExtent$addTemporalElement()
- ISOExtent$delTemporalElement()
- ISOExtent$addVerticalElement()
- ISOExtent$delVerticalElement()
- ISOExtent$clone()

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

Usage:

```
ISOExtent$new(xml = NULL)
```

Arguments:

xml	object of class XMLInternalNode-class

Method `addGeographicElement()`**:** Adds geographic element

Usage:

```
ISOExtent$addGeographicElement(element)
```

Arguments:

`element` object of class ISOGeographicExtent

Returns: `TRUE` if added, `FALSE` otherwise

Method `setGeographicElement()`**:** Sets geographic element

Usage:

```
ISOExtent$setGeographicElement(element)
```

Arguments:

`element` object of class ISOGeographicExtent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delGeographicElement()`**:** Deletes geographic element

Usage:

```
ISOExtent$delGeographicElement(element)
```

Arguments:

`element` object of class ISOGeographicExtent

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addTemporalElement()`**:** Adds temporal element

Usage:

```
ISOExtent$addTemporalElement(element)
```

Arguments:

`element` object of class ISOTemporalExtent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delTemporalElement()`**:** Deletes temporal element

Usage:

```
ISOExtent$delTemporalElement(element)
```

Arguments:

`element` object of class ISOTemporalExtent

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addVerticalElement()`**:** Adds vertical element

Usage:

```
ISOExtent$addVerticalElement(element)
```

Arguments:

`element` object of class ISOVerticalExtent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delVerticalElement()`**:** Deletes vertical element

Usage:

```
ISOExtent$delVerticalElement(element)
```

Arguments:

`element` object of class ISOVerticalExtent

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOFeatureAssociation` *ISOFeatureAssociation*

Description

ISOFeatureAssociation

ISOFeatureAssociation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOFeatureAssociation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOFeatureType` -> `ISOFeatureAssociation`

Public fields

`roleName` roleName [2..*]: ISOAssociationRole

Methods

Public methods:

- `ISOFeatureAssociation$new()`
- `ISOFeatureAssociation$addRoleName()`
- `ISOFeatureAssociation$delRoleName()`
- `ISOFeatureAssociation$clone()`

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

Usage:

```
ISOFeatureAssociation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addRoleName()`**:** Adds role name

Usage:

```
ISOFeatureAssociation$addRoleName(associationRole)
```

Arguments:

`associationRole` object of class ISOAssociationRole

Returns: `TRUE` if added, `FALSE` otherwise

Method `delRoleName()`**:** Deletes role name

Usage:

```
ISOFeatureAssociation$delRoleName(associationRole)
```

Arguments:

`associationRole` object of class ISOAssociationRole

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

ISOFeatureAttribute *ISOFeatureAttribute*

Description

ISOFeatureAttribute

ISOFeatureAttribute

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOFeatureAttribute

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractCarrierOfCharacteristics` -> `geometa::ISOAbstractPropertyType` -> `geometa::ISOPropertyType` -> `ISOFeatureAttribute`

Public fields

`code` code [0..1]: character

`valueMeasurementUnit` valueMeasurementUnit [0..1]: GMLUnitDefinition

`valueType` valueType [0..1]: ISOTypeName

`listedValue` listedValue [0..*]: ISOListedValue

Methods

Public methods:

- `ISOFeatureAttribute$new()`
- `ISOFeatureAttribute$setCode()`
- `ISOFeatureAttribute$setValueMeasurementUnit()`
- `ISOFeatureAttribute$setValueType()`
- `ISOFeatureAttribute$addListedValue()`
- `ISOFeatureAttribute$delListedValue()`
- `ISOFeatureAttribute$clone()`

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

Usage:

```
ISOFeatureAttribute$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setCode()`**:** Set code

Usage:

```
ISOFeatureAttribute$setCode(code, locales = NULL)
```

Arguments:

`code` code

`locales` list of localized codes. Default is `NULL`

Method `setValueMeasurementUnit()`**:** Set value measurement unit

Usage:

```
ISOFeatureAttribute$setValueMeasurementUnit(uom)
```

Arguments:

`uom` uom, object of class GMLUnitDefinition

Method `setValueType()`**:** Set type name

Usage:

```
ISOFeatureAttribute$setValueType(typeName, locales = NULL)
```

Arguments:

`typeName` typeName

`locales` list of localized typeNames. Default is `NULL`

Method `addListedValue()`**:** Adds listed value

Usage:

```
ISOFeatureAttribute$addListedValue(value)
```

Arguments:

`value` value, object of class ISOListedValue

Returns: `TRUE` if added, `FALSE` otherwise

Method `delListedValue()`**:** Deletes listed value

Usage:

```
ISOFeatureAttribute$delListedValue(value)
```

Arguments:

`value` value, object of class ISOListedValue

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

Examples

```
md <- ISOFeatureAttribute$new()
md$setMemberName("name")
md$setDefinition("definition")
md$setCardinality(lower=1,upper=1)
md$setCode("code")

gml <- GMLBaseUnit$new(id = "ID")
gml$setDescriptionReference("someref")
gml$setIdentifier("identifier", "codespace")
gml$addName("name1", "codespace")
gml$addName("name2", "codespace")
gml$setQuantityTypeReference("someref")
gml$setCatalogSymbol("symbol")
gml$setUnitsSystem("somelink")
md$setValueMeasurementUnit(gml)

val1 <- ISOListedValue$new()
val1$setCode("code1")
val1$setLabel("label1")
val1$setDefinition("definition1")
md$addListedValue(val1)
val2 <- ISOListedValue$new()
val2$setCode("code2")
val2$setLabel("label2")
val2$setDefinition("definition2")
md$addListedValue(val2)
md$setValueType("typeName")
```

`ISOFeatureCatalogue` *ISOFeatureCatalogue*

Description

ISOFeatureCatalogue

ISOFeatureCatalogue

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO FeatureCatalogue

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractCatalogue` -> `ISOFeatureCatalogue`

Public fields

`attrs` attrs

`producer` producer [1..1]: ISOResponsibleParty

`functionalLanguage` functionalLanguage [0..1]: character

`featureType` featureType [1..*]: ISOFeatureType

`definitionSource` definitionSource [0..*]: ISODefinitionSource

Methods

Public methods:

- `ISOFeatureCatalogue$new()`
- `ISOFeatureCatalogue$setProducer()`
- `ISOFeatureCatalogue$setFunctionalLanguage()`
- `ISOFeatureCatalogue$addFeatureType()`
- `ISOFeatureCatalogue$delFeatureType()`
- `ISOFeatureCatalogue$addDefinitionSource()`
- `ISOFeatureCatalogue$delDefinitionSource()`
- `ISOFeatureCatalogue$clone()`

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

Usage:

```
ISOFeatureCatalogue$new(xml = NULL, uuid = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`uuid` uuid

Method `setProducer()`**:** Set producer

Usage:

```
ISOFeatureCatalogue$setProducer(producer)
```

Arguments:

`producer` object of class ISOResponsibleParty

Method `setFunctionalLanguage()`**:** Set functional language

Usage:

```
ISOFeatureCatalogue$setFunctionalLanguage(functionalLanguage)
```

Arguments:

`functionalLanguage` functional language

Method `addFeatureType()`**:** Adds feature type

Usage:

```
ISOFeatureCatalogue$addFeatureType(featureType)
```

Arguments:

`featureType` object of class ISOFeatureType

Returns: `TRUE` if added, `FALSE` otherwise

Method `delFeatureType()`**:** Deletes feature type

Usage:

```
ISOFeatureCatalogue$delFeatureType(featureType)
```

Arguments:

`featureType` object of class ISOFeatureType

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addDefinitionSource()`**:** Adds definition source

Usage:

```
ISOFeatureCatalogue$addDefinitionSource(source)
```

Arguments:

`source` object of class ISODefinitionSource or ISOCitation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDefinitionSource()`**:** Deletes definition source

Usage:

```
ISOFeatureCatalogue$delDefinitionSource(source)
```

Arguments:

`source` object of class ISODefinitionSource or ISOCitation

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

Examples

```
fc <- ISOFeatureCatalogue$new(uuid = "my-fc-identifier")
fc$setName("name")
fc$addScope("scope1")
fc$addScope("scope2")
fc$addFieldOfApplication("field1")
fc$addFieldOfApplication("field2")
fc$setVersionNumber("1.0")
fc$setVersionDate(ISOdate(2015, 1, 1, 1))

producer <- ISOResponsibleParty$new()
producer$setIndividualName("someone")
fc$setProducer(producer)
fc$setFunctionalLanguage("eng")

cit <- ISOCitation$new()
cit$setTitle("some citation title")
fc$addDefinitionSource(cit)
#'  #add featureType
ft <- ISOFeatureType$new()
ft$setTypeName("typeName")
ft$setDefinition("definition")
ft$setCode("code")
ft$setIsAbstract(FALSE)
ft$addAlias("alias1")
ft$addAlias("alias2")

#add feature attributes
for(i in 1:3){
  #create attribute
  fat <- ISOFeatureAttribute$new()
  fat$setMemberName(sprintf("name %s",i))
  fat$setDefinition(sprintf("definition %s",i))
  fat$setCardinality(lower=1,upper=1)
  fat$setCode(sprintf("code %s",i))

  gml <- GMLBaseUnit$new(id = sprintf("ID%s",i))
  gml$setDescriptionReference("someref")
  gml$setIdentifier("identifier", "codespace")
  gml$addName("name1", "codespace")
  gml$addName("name2", "codespace")
  gml$setQuantityTypeReference("someref")
  gml$setCatalogSymbol("symbol")
  gml$setUnitsSystem("somelink")
  fat$setValueMeasurementUnit(gml)

  #add listed values
  val1 <- ISOListedValue$new()
  val1$setCode("code1")
  val1$setLabel("label1")
  val1$setDefinition("definition1")
  fat$addListedValue(val1)
```

```
    val2 <- ISOListedValue$new()
    val2$setCode("code2")
    val2$setLabel("label2")
    val2$setDefinition("definition2")
    fat$addListedValue(val2)
    fat$setValueType("typeName")

    #add feature attribute as carrierOfCharacteristic
    ft$addCharacteristic(fat)
  }
  #add featureType to catalogue
  fc$addFeatureType(ft)

  xml <- fc$encode()
```

`ISOFeatureCatalogueDescription` *ISOFeatureCatalogueDescription*

Description

ISOFeatureCatalogueDescription

ISOFeatureCatalogueDescription

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOFeatureCatalogue

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOContentInformation` -> `ISOFeatureCatalogueDescription`

Public fields

`complianceCode` complianceCode: logical

`language` language [0..*]: character

`includedWithDataset` includedWithDataset: logical

`featureTypes` featureTypes [0..*]: GenericName #TODO?

`featureCatalogueCitation` featureCatalogueCitation [1..*]: ISOCitation

Methods

Public methods:

- `ISOFeatureCatalogueDescription$new()`
- `ISOFeatureCatalogueDescription$setComplianceCode()`
- `ISOFeatureCatalogueDescription$addLanguage()`
- `ISOFeatureCatalogueDescription$delLanguage()`
- `ISOFeatureCatalogueDescription$setIncludedWithDataset()`
- `ISOFeatureCatalogueDescription$addFeatureCatalogueCitation()`
- `ISOFeatureCatalogueDescription$delFeatureCatalogueCitation()`
- `ISOFeatureCatalogueDescription$clone()`

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

Usage:

```
ISOFeatureCatalogueDescription$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setComplianceCode()`**:** Set compliance code

Usage:

```
ISOFeatureCatalogueDescription$setComplianceCode(compliance)
```

Arguments:

`compliance` compliance, object of class logical

Method `addLanguage()`**:** Adds language

Usage:

```
ISOFeatureCatalogueDescription$addLanguage(lang)
```

Arguments:

`lang` lang

Returns: `TRUE` if added, `FALSE` otherwise

Method `delLanguage()`**:** Deletes language

Usage:

```
ISOFeatureCatalogueDescription$delLanguage(lang)
```

Arguments:

`lang` lang

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setIncludedWithDataset()`**:** Set included with dataset

Usage:

```
ISOFeatureCatalogueDescription$setIncludedWithDataset(include)
```

Arguments:

`include` include, object of class logical

Method `addFeatureCatalogueCitation()`**:** Adds feature catalogue citation

Usage:

```
ISOFeatureCatalogueDescription$addFeatureCatalogueCitation(
  citation,
  uuid = NULL
)
```

Arguments:

`citation,` object of class ISOCitation

`uuid` uuid

Returns: `TRUE` if added, `FALSE` otherwise

Method `delFeatureCatalogueCitation()`**:** Deletes feature catalogue citation

Usage:

```
ISOFeatureCatalogueDescription$delFeatureCatalogueCitation(
  citation,
  uuid = NULL
)
```

Arguments:

`citation,` object of class ISOCitation

`uuid` uuid

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOFeatureCatalogueDescription$new()
md$setComplianceCode(FALSE)
md$addLanguage("eng")
md$setIncludedWithDataset(FALSE)

cit = ISOCitation$new()
```

```
contact = ISOContact$new()
fcLink <- ISOOnlineResource$new()
fcLink$setLinkage("http://somelink/featurecatalogue")
contact$setOnlineResource(fcLink)
rp = ISOResponsibleParty$new()
rp$setContactInfo(contact)
cit$setCitedResponsibleParty(rp)
md$addFeatureCatalogueCitation(cit)
```

`ISOFeatureOperation` *ISOFeatureOperation*

Description

ISOFeatureOperation

ISOFeatureOperation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOFeatureOperation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractCarrierOfCharacteristics` -> `geometa::ISOAbstractPropertyType` -> `geometa::ISOPropertyType` -> `ISOFeatureOperation`

Public fields

`signature` signature: character

`formalDefinition` formalDefinition [0..1]: character

Methods

Public methods:

- `ISOFeatureOperation$new()`
- `ISOFeatureOperation$setSignature()`
- `ISOFeatureOperation$setFormalDefinition()`
- `ISOFeatureOperation$clone()`

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

Usage:

```
ISOFeatureOperation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setSignature()`**:** Set signature

Usage:

```
ISOFeatureOperation$setSignature(signature, locales = NULL)
```

Arguments:

`signature` signature

`locales` list of localized signatures. Default is `NULL`

Method `setFormalDefinition()`**:** Set formal definition

Usage:

```
ISOFeatureOperation$setFormalDefinition(formalDefinition, locales = NULL)
```

Arguments:

`formalDefinition` formal definition

`locales` list of localized definitions. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

Examples

```
md <- ISOFeatureOperation$new()
md$setMemberName("name")
md$setDefinition("definition")
md$setCardinality(lower=1,upper=1)
md$setSignature("signature")
md$setFormalDefinition("def")
```

`ISOFeatureType` *ISOFeatureType*

Description

ISOFeatureType

ISOFeatureType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO FeatureType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOFeatureType`

Public fields

`typeName` typeName [1..1]: ISOLocalName

`definition` definition [0..1]: character

`code` code [0..1]: character

`isAbstract` isAbstract [1..1]: logical

`aliases` aliases [0..*]: ISOLocalName

`inheritsFrom` inheritsFrom [0..*]: ISOInheritanceRelation

`inheritsTo` inheritsTo [0..*]: ISOInheritanceRelation

`featureCatalogue` featureCatalogue: ISOFeatureCatalogue

`constrainedBy` constrainedBy [0..*]: ISOConstraint

`definitionReference` definitionReference [0..*]: ISODefinitionReference

`carrierOfCharacteristics` carrierOfCharacteristics [0..*]: ISOCarrierOfCharacteristics

Methods

Public methods:

- `ISOFeatureType$new()`
- `ISOFeatureType$setTypeName()`
- `ISOFeatureType$setDefinition()`
- `ISOFeatureType$setCode()`
- `ISOFeatureType$setIsAbstract()`
- `ISOFeatureType$addAlias()`

- `ISOFeatureType$delAlias()`
- `ISOFeatureType$addInheritsFrom()`
- `ISOFeatureType$delInheritsFrom()`
- `ISOFeatureType$addInheritsTo()`
- `ISOFeatureType$delInheritsTo()`
- `ISOFeatureType$setFeatureCatalogue()`
- `ISOFeatureType$addConstraint()`
- `ISOFeatureType$delConstraint()`
- `ISOFeatureType$setDefinitionReference()`
- `ISOFeatureType$addCharacteristic()`
- `ISOFeatureType$delCharacteristic()`
- `ISOFeatureType$clone()`

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

Usage:

```
ISOFeatureType$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setTypeName()`**:** Set type name

Usage:

```
ISOFeatureType$setTypeName(typeName)
```

Arguments:

`typeName` type name, object of class ISOLocalName or character

Method `setDefinition()`**:** Set definition

Usage:

```
ISOFeatureType$setDefinition(definition, locales = NULL)
```

Arguments:

`definition` definition

`locales` list of localized definitions. Default is `NULL`

Method `setCode()`**:** Set code

Usage:

```
ISOFeatureType$setCode(code, locales = NULL)
```

Arguments:

`code` definition

`locales` list of localized codes. Default is `NULL`

Method `setIsAbstract()`**:** Set whether feature type is abstract

Usage:

```
ISOFeatureType$setIsAbstract(isAbstract)
```

Arguments:

`isAbstract` object of class logical

Method `addAlias()`**:** Adds alias

Usage:

```
ISOFeatureType$addAlias(alias)
```

Arguments:

`alias` object of class ISOLocalName or character

Returns: `TRUE` if added, `FALSE` otherwise

Method `delAlias()`**:** Deletes alias

Usage:

```
ISOFeatureType$delAlias(alias)
```

Arguments:

`alias` object of class ISOLocalName or character

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addInheritsFrom()`**:** Adds 'inheritsFrom' relation

Usage:

```
ISOFeatureType$addInheritsFrom(rel)
```

Arguments:

`rel` rel, object of class ISOInheritanceRelation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delInheritsFrom()`**:** Deletes 'inheritsFrom' relation

Usage:

```
ISOFeatureType$delInheritsFrom(rel)
```

Arguments:

`rel` rel, object of class ISOInheritanceRelation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addInheritsTo()`**:** Adds 'inheritsTo' relation

Usage:

```
ISOFeatureType$addInheritsTo(rel)
```

Arguments:

`rel` rel, object of class ISOInheritanceRelation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delInheritsTo()`**:** Deletes 'inheritsTo' relation

Usage:

```
ISOFeatureType$delInheritsTo(rel)
```

Arguments:

`rel` rel, object of class ISOInheritanceRelation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setFeatureCatalogue()`**:** Set feature catalogue

Usage:

```
ISOFeatureType$setFeatureCatalogue(fc)
```

Arguments:

`fc` object of class ISOFeatureCatalogue

Method `addConstraint()`**:** Adds constraint

Usage:

```
ISOFeatureType$addConstraint(constraint)
```

Arguments:

`constraint` constraint, object of class ISOConstraint

Returns: `TRUE` if added, `FALSE` otherwise

Method `delConstraint()`**:** Deletes constraint

Usage:

```
ISOFeatureType$delConstraint(constraint)
```

Arguments:

`constraint` constraint, object of class ISOConstraint

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setDefinitionReference()`**:** Set definition reference

Usage:

```
ISOFeatureType$setDefinitionReference(definitionReference)
```

Arguments:

`definitionReference` object of class ISODefinitionReference

Method `addCharacteristic()`**:** Adds characteristic

Usage:

```
ISOFeatureType$addCharacteristic(characteristic)
```

Arguments:

`characteristic` characteristic, object inheriting class ISOAbstractCarrierOfCharacteristics

Returns: `TRUE` if added, `FALSE` otherwise

Method `delCharacteristic()`**:** Deletes characteristic

Usage:

```
ISOFeatureType$delCharacteristic(characteristic)
```

Arguments:

`characteristic` characteristic, object inheriting class ISOAbstractCarrierOfCharacteristics

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOFeatureType$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

Examples

```
#featuretype
md <- ISOFeatureType$new()
md$setTypeName("typeName")
md$setDefinition("definition")
md$setCode("code")
md$setIsAbstract(FALSE)
md$addAlias("alias1")
md$addAlias("alias2")

#add feature attributes
for(i in 1:3){
  #create attribute
  fat <- ISOFeatureAttribute$new()
  fat$setMemberName(sprintf("name %s",i))
  fat$setDefinition(sprintf("definition %s",i))
  fat$setCardinality(lower=1,upper=1)
  fat$setCode(sprintf("code %s",i))

  #add measurement unit
  gml <- GMLBaseUnit$new(id = "ID%")
  gml$setDescriptionReference("someref")
  gml$setIdentifier("identifier", "codespace")
  gml$addName("name1", "codespace")
  gml$addName("name2", "codespace")
  gml$setQuantityTypeReference("someref")
  gml$setCatalogSymbol("symbol")
  gml$setUnitsSystem("somelink")
  fat$setValueMeasurementUnit(gml)

  #add listed values
  val1 <- ISOListedValue$new()
  val1$setCode("code1")
  val1$setLabel("label1")
```

```
    val1$setDefinition("definition1")
    fat$addListedValue(val1)
    val2 <- ISOListedValue$new()
    val2$setCode("code2")
    val2$setLabel("label2")
    val2$setDefinition("definition2")
    fat$addListedValue(val2)
    fat$setValueType("typeName")

    #add feature attribute as carrierOfCharacteristic
    md$addCharacteristic(fat)
  }
  xml <- md$encode()
```

`ISOFileName` *ISOFileName*

Description

ISOFileName

ISOFileName

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO FileName

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOFileName`

Public fields

`attrs` attrs

Methods

Public methods:

- `ISOFileName$new()`
- `ISOFileName$clone()`

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

Usage:

```
ISOFileName$new(xml = NULL, file = NULL, name = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`file` file

`name` name

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

Usage:

`ISOFileName$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19139:2007 Geographic information – XML

Examples

```
md <- ISOFileName$new(file = "someuri", name = "filename")
xml <- md$encode()
```

`ISOFormat` *ISOFormat*

Description

ISOFormat

ISOFormat

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOFormat

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOFormat`

Public fields

`name` name : CharacterString

`version` version : CharacterString

`amendmentNumber` amendmentNumber [0..1] : CharacterString

`specification` specification [0..1] : CharacterString

`fileDecompressionTechnique` fileDecompressionTechnique [0..1] : CharacterString

`FormatDistributor` FormatDistributor [0..*]: ISODistributor

Methods

Public methods:

- `ISOFormat$new()`
- `ISOFormat$setName()`
- `ISOFormat$setVersion()`
- `ISOFormat$setAmendmentNumber()`
- `ISOFormat$setSpecification()`
- `ISOFormat$setFileDecompressionTechnique()`
- `ISOFormat$addDistributor()`
- `ISOFormat$delDistributor()`
- `ISOFormat$clone()`

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

Usage:

```
ISOFormat$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setName()`**:** Set name

Usage:

```
ISOFormat$setName(name, locales = NULL)
```

Arguments:

`name` name

`locales` list of localized names. Default is `NULL`

Method `setVersion()`**:** Set version

Usage:

```
ISOFormat$setVersion(version)
```

Arguments:

`version` version

Method `setAmendmentNumber()`**:** Set amendment number

Usage:

`ISOFormat$setAmendmentNumber(amendmentNumber)`

Arguments:

`amendmentNumber` amendment number

Method `setSpecification():` Set specification

Usage:

`ISOFormat$setSpecification(specification, locales = NULL)`

Arguments:

`specification` specification

`locales` list of localized specifications. Default is `NULL`

Method `setFileDecompressionTechnique():` Set file decompression technique

Usage:

`ISOFormat$setFileDecompressionTechnique(technique)`

Arguments:

`technique` technique

Method `addDistributor():` Adds distributor

Usage:

`ISOFormat$addDistributor(distributor)`

Arguments:

`distributor` object of class ISODistributor

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDistributor():` Deletes distributor

Usage:

`ISOFormat$delDistributor(distributor)`

Arguments:

`distributor` object of class ISODistributor

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOFormat$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOFormat$new()
md$setName("name")
md$setVersion("1.0")
md$setAmendmentNumber("2")
md$setSpecification("specification")
```

`ISOFormatConsistency` *ISOFormatConsistency*

Description

ISOFormatConsistency

ISOFormatConsistency

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOFormatConsistency

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `geometa::ISOAbstractLogicalConsistency` -> `ISOFormatConsistency`

Methods

Public methods:

- `ISOFormatConsistency$clone()`

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

Usage:

`ISOFormatConsistency$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOFormatConsistency$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

`ISOFreeText` *ISOFreeText*

Description

ISOFreeText

ISOFreeText

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO FreeText

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOFreeText`

Public fields

`textGroup` textGroup [1..*]: ISOLocalisedCharacterString

Methods

Public methods:

- `ISOFreeText$new()`
- `ISOFreeText$addTextGroup()`
- `ISOFreeText$delTextGroup()`
- `ISOFreeText$clone()`

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

Usage:

```
ISOFreeText$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addTextGroup()`**:** Adds text group

Usage:

```
ISOFreeText$addTextGroup(textGroup)
```

Arguments:

`textGroup` text group, object of class ISOLocalisedCharacterString

Returns: `TRUE` if added, `FALSE` otherwise

Method `delTextGroup()`**:** Deletes text group

Usage:

```
ISOFreeText$delTextGroup(textGroup)
```

Arguments:

`textGroup` text group, object of class ISOLocalisedCharacterString

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
ft <- ISOFreeText$new()
```

`ISOGeographicBoundingBox` *ISOGeographicBoundingBox*

Description

ISOGeographicBoundingBox

ISOGeographicBoundingBox

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO GeographicBoundingBox

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOGeographicExtent` -> `ISOGeographicBoundingBox`

Public fields

`westBoundLongitude`	westBoundLongitude
`eastBoundLongitude`	eastBoundLongitude
`southBoundLatitude`	southBoundLatitude
`northBoundLatitude`	northBoundLatitude

Methods

Public methods:

- `ISOGeographicBoundingBox$new()`
- `ISOGeographicBoundingBox$setWestBoundLongitude()`
- `ISOGeographicBoundingBox$setEastBoundLongitude()`
- `ISOGeographicBoundingBox$setSouthBoundLatitude()`
- `ISOGeographicBoundingBox$setNorthBoundLatitude()`
- `ISOGeographicBoundingBox$clone()`

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

Usage:

```
ISOGeographicBoundingBox$new(
  xml = NULL,
  minx = NULL,
  miny = NULL,
  maxx = NULL,
  maxy = NULL,
  bbox = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`minx` minx object of class numeric

`miny` miny object of class numeric

`maxx` maxx object of class numeric

`maxy` maxy object of class numeric

`bbox` bbox object of class matrix

Method `setWestBoundLongitude()`**:** Set west bound longitude

Usage:

`ISOGeographicBoundingBox$setWestBoundLongitude(minx)`

Arguments:

`minx` minx object of class numeric

Method `setEastBoundLongitude()`**:** Set east bound longitude

Usage:

`ISOGeographicBoundingBox$setEastBoundLongitude(maxx)`

Arguments:

`maxx` maxx object of class numeric

Method `setSouthBoundLatitude()`**:** Set south bound latitude

Usage:

`ISOGeographicBoundingBox$setSouthBoundLatitude(miny)`

Arguments:

`miny` miny object of class numeric

Method `setNorthBoundLatitude()`**:** Set north bound latitude

Usage:

`ISOGeographicBoundingBox$setNorthBoundLatitude(maxy)`

Arguments:

`maxy` maxy object of class numeric

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

Usage:

`ISOGeographicBoundingBox$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOGeographicBoundingBox$new(minx = -180, miny = -90, maxx = 180, maxy = 90)
xml <- md$encode()
```

ISOGeographicDescription
ISOGeographicDescription

Description

ISOGeographicDescription

ISOGeographicDescription

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO GeographicDescription

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOGeographicExtent` -> `ISOGeographicDescription`

Public fields

`geographicIdentifier` geographicIdentifier [1..1]: character

Methods

Public methods:

- `ISOGeographicDescription$new()`
- `ISOGeographicDescription$setGeographicIdentifier()`
- `ISOGeographicDescription$clone()`

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

Usage:

```
ISOGeographicDescription$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setGeographicIdentifier()`**:** Set geographic identifier

Usage:

```
ISOGeographicDescription$setGeographicIdentifier(geographicIdentifier)
```

Arguments:

`geographicIdentifier` geographic identifier, object of class ISOMetaIdentifier

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOGeographicDescription$new()
md$setGeographicIdentifier(ISOMetaIdentifier$new(code = "identifier"))
xml <- md$encode()
```

`ISOGeographicExtent` *ISOGeographicExtent*

Description

ISOGeographicExtent

ISOGeographicExtent

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO abstract geographicExtent

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOGeographicExtent`

Public fields

`extentTypeCode` extentTypeCode [0..1]: ISOBaseBoolean default "true"

Methods

Public methods:

- `ISOGeographicExtent$new()`
- `ISOGeographicExtent$clone()`

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

Usage:

```
ISOGeographicExtent$new(xml = NULL, defaults = list())
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` defaults list

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

abstract class

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOGeometricObjects` *ISOGeometricObjects*

Description

ISOGeometricObjects

ISOGeometricObjects

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO GeometricObjects

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOGeometricObjects`

Public fields

`geometricObjectType` geometricObjectType

`geometricObjectCount` geometricObjectCount

Methods

Public methods:

- `ISOGeometricObjects$new()`
- `ISOGeometricObjects$setGeometricObjectType()`
- `ISOGeometricObjects$setGeometricObjectCount()`
- `ISOGeometricObjects$clone()`

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

Usage:

```
ISOGeometricObjects$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setGeometricObjectType()`**:** Set geometric object type

Usage:

```
ISOGeometricObjects$setGeometricObjectType(geometricObjectType)
```

Arguments:

`geometricObjectType` object of class ISOGeometricObjectType or any character among values returned by `ISOGeometricObjectType$values()`

Method `setGeometricObjectCount()`**:** Set geometric object count

Usage:

`ISOGeometricObjects$setGeometricObjectCount(geometricObjectCount)`

Arguments:

`geometricObjectCount` object of class integer

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

Usage:

`ISOGeometricObjects$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOGeometricObjects$new()
md$setGeometricObjectType("surface")
md$setGeometricObjectCount(5L)
xml <- md$encode()
```

`ISOGeometricObjectType` *ISOGeometricObjectType*

Description

ISOGeometricObjectType

ISOGeometricObjectType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO GeometricObjectType

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISOCodeListValue -> ISOGeometricObjectType

Methods

Public methods:

- ISOGeometricObjectType$new()
- ISOGeometricObjectType$clone()

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

Usage:

```
ISOGeometricObjectType$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOGeometricObjectType$values(labels = TRUE)

#point type
pt <- ISOGeometricObjectType$new(value = "point")
```

`ISOGeorectified` *ISOGeorectified*

Description

ISOGeorectified

ISOGeorectified

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Georectified

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOSpatialRepresentation` -> `geometa::ISOGridSpatialRepresentation` -> `ISOGeorectified`

Public fields

`checkPointAvailability` checkPointAvailability [1..1]

`checkPointDescription` checkPointDescription [0..1]

`cornerPoints` cornerPoints [0..*]

`centerPoint` centerPoint [0..1]

`pointInPixel` pointInPixel [1..1]

`transformationDimensionDescription` transformationDimensionDescription [0..1]

`transformationDimensionMapping` transformationDimensionMapping [0..2]

Methods

Public methods:

- `ISOGeorectified$new()`
- `ISOGeorectified$setCheckPointAvailability()`
- `ISOGeorectified$setCheckPointDescription()`
- `ISOGeorectified$addCornerPoint()`
- `ISOGeorectified$delCornerPoint()`
- `ISOGeorectified$setCenterPoint()`
- `ISOGeorectified$setPixelOrientation()`
- `ISOGeorectified$setTransformationDimensionDescription()`
- `ISOGeorectified$addTransformationDimensionMapping()`
- `ISOGeorectified$delTransformationDimensionMapping()`

- `ISOGeorectified$clone()`

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

Usage:

```
ISOGeorectified$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setCheckPointAvailability()`**:** Set check point availability

Usage:

```
ISOGeorectified$setCheckPointAvailability(availability)
```

Arguments:

`availability` object of class logical

Method `setCheckPointDescription()`**:** Set check point description

Usage:

```
ISOGeorectified$setCheckPointDescription(description, locales = NULL)
```

Arguments:

`description` object of class character

`locales` list of localized descriptions. Default is `NULL`

Method `addCornerPoint()`**:** Adds corner point

Usage:

```
ISOGeorectified$addCornerPoint(sfg = NULL, m = NULL)
```

Arguments:

`sfg` simple feature object from **sf**

`m` simple feature object of class matrix

Returns: `TRUE` if added, `FALSE` otherwise

Method `delCornerPoint()`**:** Deletes corner point

Usage:

```
ISOGeorectified$delCornerPoint(sfg = NULL, m = NULL)
```

Arguments:

`sfg` simple feature object from **sf**

`m` simple feature object of class matrix

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setCenterPoint()`**:** Sets center point

Usage:

```
ISOGeorectified$setCenterPoint(sfg = NULL, m = NULL)
```

Arguments:

`sfg` simple feature object from **sf**

`m` simple feature object of class matrix

Method `setPixelOrientation()`**:** Set pixel orientation

Usage:

```
ISOGeorectified$setPixelOrientation(pixelOrientation)
```

Arguments:

`pixelOrientation` object of class ISOPixelOrientation or character among values among those returned by `ISOPixelOrientation$values()`

Method `setTransformationDimensionDescription()`**:** Set transformation dimension description

Usage:

```
ISOGeorectified$setTransformationDimensionDescription(
  description,
  locales = NULL
)
```

Arguments:

`description` description

`locales` list of localized descriptions. Default is `NULL`

Method `addTransformationDimensionMapping()`**:** Adds transformation dimension mapping

Usage:

```
ISOGeorectified$addTransformationDimensionMapping(mapping)
```

Arguments:

`mapping` mapping

Returns: `TRUE` if added, `FALSE` otherwise

Method `delTransformationDimensionMapping()`**:** Deletes transformation dimension mapping

Usage:

```
ISOGeorectified$delTransformationDimensionMapping(mapping)
```

Arguments:

`mapping` mapping

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOGeoreferenceable` *ISOGeoreferenceable*

Description

ISOGeoreferenceable

ISOGeoreferenceable

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Georeferenceable

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOSpatialRepresentation` -> `geometa::ISOGridSpatialRepresentation` -> `ISOGeoreferenceable`

Public fields

`controlPointAvailability` controlPointAvailability: logical

`orientationParameterAvailability` orientationParameterAvailability : logical

`orientationParameterDescription` orientationParameterDescription [0..1] : character

`georeferencedParameters` georeferencedParameters : ISORecord

`parameterCitation` parameterCitation [0..*] : ISOCitation

Methods

Public methods:

- `ISOGeoreferenceable$new()`
- `ISOGeoreferenceable$setControlPointAvailability()`
- `ISOGeoreferenceable$setOrientationParameterAvailability()`
- `ISOGeoreferenceable$setOrientationParameterDescription()`
- `ISOGeoreferenceable$setGeoreferencedParameters()`
- `ISOGeoreferenceable$addParameterCitation()`
- `ISOGeoreferenceable$delParameterCitation()`
- `ISOGeoreferenceable$clone()`

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

Usage:

```
ISOGeoreferenceable$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setControlPointAvailability()`**:** Set control point availability

Usage:

```
ISOGeoreferenceable$setControlPointAvailability(availability)
```

Arguments:

`availability` object of class logical

Method `setOrientationParameterAvailability()`**:** Set orientation parameter availability

Usage:

```
ISOGeoreferenceable$setOrientationParameterAvailability(availability)
```

Arguments:

`availability` object of class logical

Method `setOrientationParameterDescription()`**:** Set orientation parameter description

Usage:

```
ISOGeoreferenceable$setOrientationParameterDescription(
  description,
  locales = NULL
)
```

Arguments:

`description` description

`locales` list of localized descriptions. Default is `NULL`

Method `setGeoreferencedParameters()`**:** Set georeferenced parameters

Usage:

```
ISOGeoreferenceable$setGeoreferencedParameters(record)
```

Arguments:

`record` object of class ISORecord

Method `addParameterCitation()`**:** Adds parameter citation

Usage:

```
ISOGeoreferenceable$addParameterCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delParameterCitation()`**:** Deletes parameter citation

Usage:

```
ISOGeoreferenceable$delParameterCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOGeoreferenceable$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOGeoreferenceable$new()

#inherited methods from ISOGridSpatialRepresentation
md$setNumberOfDimensions(1)
dim1 <- ISODimension$new()
dim1$setName("row")
dim1$setSize(100)
dim1$setResolution(ISOMeasure$new(value=1,uom="m"))
md$addDimension(dim1)
md$setCellGeometry("area")

#parameters
md$setControlPointAvailability(TRUE)
md$setOrientationParameterAvailability(TRUE)
md$setOrientationParameterDescription("description")
md$setGeoreferencedParameters("record")
ct <- ISOCitation$new()
ct$setTitle("citation")
md$addParameterCitation(ct)

xml <- md$encode()
```

`ISOGriddedDataPositionalAccuracy`
ISOGriddedDataPositionalAccuracy

Description

ISOGriddedDataPositionalAccuracy

ISOGriddedDataPositionalAccuracy

Format

R6Class object.

Value

Object of R6Class for modelling an ISOGriddedDataPositionalAccuracy

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `geometa::ISOAbstractPositionalAccuracy` -> `ISOGriddedDataPositionalAccuracy`

Methods

Public methods:

- `ISOGriddedDataPositionalAccuracy$clone()`

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

Usage:

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

Arguments:

deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOGriddedDataPositionalAccuracy$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
```

```
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

ISOGridSpatialRepresentation *ISOGridSpatialRepresentation*

Description

ISOGridSpatialRepresentation

ISOGridSpatialRepresentation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO GridSpatialRepresentation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOSpatialRepresentation` -> `ISOGridSpatialRepresentation`

Public fields

`numberOfDimensions` numberOfDimensions [1..1]: integer

`axisDimensionProperties` axisDimensionProperties [1..*] : ISODimension

`cellGeometry` cellGeometry [1..1]: ISOCellGeometry

`transformationParameterAvailability` transformationParameterAvailability : logical

Methods

Public methods:

- `ISOGridSpatialRepresentation$new()`
- `ISOGridSpatialRepresentation$setNumberOfDimensions()`
- `ISOGridSpatialRepresentation$addDimension()`
- `ISOGridSpatialRepresentation$delDimension()`
- `ISOGridSpatialRepresentation$setCellGeometry()`

- `ISOGridSpatialRepresentation$setTransformationParameterAvailability()`
- `ISOGridSpatialRepresentation$clone()`

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

Usage:

```
ISOGridSpatialRepresentation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setNumberOfDimensions()`**:** Set number of dimensions

Usage:

```
ISOGridSpatialRepresentation$setNumberOfDimensions(numberOfDimensions)
```

Arguments:

`numberOfDimensions` object of class integer

Method `addDimension()`**:** Adds dimension

Usage:

```
ISOGridSpatialRepresentation$addDimension(dimension)
```

Arguments:

`dimension` object of class ISODimension

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDimension()`**:** Deletes dimension

Usage:

```
ISOGridSpatialRepresentation$delDimension(dimension)
```

Arguments:

`dimension` object of class ISODimension

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setCellGeometry()`**:** Set cell geometry

Usage:

```
ISOGridSpatialRepresentation$setCellGeometry(cellGeometry)
```

Arguments:

`cellGeometry` object of class ISOCellGeometry or any character among values returned by `ISOCellGeometry$values()`

Method `setTransformationParameterAvailability()`**:** Set transformation parameter availability

Usage:

```
ISOGridSpatialRepresentation$setTransformationParameterAvailability(
  availability
)
```

Arguments:

`availability` object of class logical

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

Usage:

`ISOGridSpatialRepresentation$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOGridSpatialRepresentation$new()
md$setNumberOfDimensions(1)
dim1 <- ISODimension$new()
dim1$setName("row")
dim1$setSize(100)
dim1$setResolution(ISOMeasure$new(value=1,uom="m"))
md$addDimension(dim1)
md$setCellGeometry("area")
xml <- md$encode()
```

`ISOHierarchyLevel` *ISOHierarchyLevel*

Description

ISOHierarchyLevel

ISOHierarchyLevel

Format

`R6Class` object

Value

Object of `R6Class` for modelling an ISO HierarchyLevel

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOHierarchyLevel`

Methods

Public methods:

- `ISOHierarchyLevel$new()`
- `ISOHierarchyLevel$clone()`

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

Usage:

```
ISOHierarchyLevel$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOHierarchyLevel$values(labels = TRUE)

#dataset scope
ds <- ISOHierarchyLevel$new(value = "dataset")
```

`ISOIdentification` *ISOIdentification*

Description

ISOIdentification

ISOIdentification

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Identification

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOIdentification`

Public fields

`citation` citation: ISOCitation

`abstract` abstract: character

`purpose` purpose [0..1]: character

`credit` credit [0..*]: character

`status` status [0..*]: ISOStatus

`pointOfContact` pointOfContact [0..*]: ISOResponsibleParty

`resourceMaintenance` resourceMaintenance [0..*]: ISOMaintenanceInformation

`graphicOverview` graphicOverview [0..*]: ISOBrowseGraphic

`resourceFormat` resourceFormat [0..*]: ISOFormat

`descriptiveKeywords` descriptiveKeywords [0..*]: ISOKeywords

`resourceConstraints` resourceConstraints [0..*]: ISOLegalConstraints

`resourceSpecificUsage` resourceSpecificUsage [0..*]: MD_Usage (ISOUsage - to implement)

`aggregationInfo` aggregationInfo [0..*]: ISOAggregateInformation

Methods

Public methods:

- `ISOIdentification$new()`
- `ISOIdentification$setCitation()`
- `ISOIdentification$setAbstract()`
- `ISOIdentification$setPurpose()`
- `ISOIdentification$addCredit()`
- `ISOIdentification$delCredit()`
- `ISOIdentification$addStatus()`
- `ISOIdentification$delStatus()`
- `ISOIdentification$addPointOfContact()`
- `ISOIdentification$delPointOfContact()`
- `ISOIdentification$addResourceMaintenance()`
- `ISOIdentification$setResourceMaintenance()`
- `ISOIdentification$delResourceMaintenance()`

- `ISOIdentification$addGraphicOverview()`
- `ISOIdentification$setGraphicOverview()`
- `ISOIdentification$delGraphicOverview()`
- `ISOIdentification$addFormat()`
- `ISOIdentification$delFormat()`
- `ISOIdentification$addKeywords()`
- `ISOIdentification$setKeywords()`
- `ISOIdentification$delKeywords()`
- `ISOIdentification$addResourceConstraints()`
- `ISOIdentification$setResourceConstraints()`
- `ISOIdentification$delResourceConstraints()`
- `ISOIdentification$addAggregateInformation()`
- `ISOIdentification$delAggregateInformation()`
- `ISOIdentification$clone()`

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

Usage:

```
ISOIdentification$new(xml = NULL, defaults = list())
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` defaults list

Method `setCitation()`**:** Set citation

Usage:

```
ISOIdentification$setCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Method `setAbstract()`**:** Set abstract

Usage:

```
ISOIdentification$setAbstract(abstract, locales = NULL)
```

Arguments:

`abstract` abstract

`locales` list of localized abstracts. Default is `NULL`

Method `setPurpose()`**:** Set purpose

Usage:

```
ISOIdentification$setPurpose(purpose, locales = NULL)
```

Arguments:

`purpose` purpose

`locales` list of localized texts. Default is `NULL`

Method `addCredit()`**:** Adds credit

Usage:

```
ISOIdentification$addCredit(credit, locales = NULL)
```

Arguments:

`credit` credit

`locales` list of localized texts. Default is `NULL`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delCredit()`**:** Deletes credit

Usage:

```
ISOIdentification$delCredit(credit, locales = NULL)
```

Arguments:

`credit` credit

`locales` list of localized texts. Default is `NULL`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addStatus()`**:** Adds status

Usage:

```
ISOIdentification$addStatus(status)
```

Arguments:

`status` object of class ISOStatus or any character among values returned by `ISOStatus$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delStatus()`**:** Deletes status

Usage:

```
ISOIdentification$delStatus(status)
```

Arguments:

`status` object of class ISOStatus or any character among values returned by `ISOStatus$values()`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addPointOfContact()`**:** Adds point of contact

Usage:

```
ISOIdentification$addPointOfContact(pointOfContact)
```

Arguments:

`pointOfContact` object of class ISOResponsibleParty

Returns: `TRUE` if added, `FALSE` otherwise

Method `delPointOfContact()`**:** Deletes point of contact

Usage:

```
ISOIdentification$delPointOfContact(pointOfContact)
```

Arguments:

`pointOfContact` object of class ISOResponsibleParty

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addResourceMaintenance()`**:** Adds resource maintenance

Usage:

```
ISOIdentification$addResourceMaintenance(resourceMaintenance)
```

Arguments:

`resourceMaintenance` object of class ISOMaintenanceInformation

Returns: `TRUE` if added, `FALSE` otherwise

Method `setResourceMaintenance()`**:** Set resource maintenance

Usage:

```
ISOIdentification$setResourceMaintenance(resourceMaintenance)
```

Arguments:

`resourceMaintenance` object of class ISOMaintenanceInformation

Returns: `TRUE` if set, `FALSE` otherwise

Method `delResourceMaintenance()`**:** Deletes resource maintenance

Usage:

```
ISOIdentification$delResourceMaintenance(resourceMaintenance)
```

Arguments:

`resourceMaintenance` object of class ISOMaintenanceInformation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addGraphicOverview()`**:** Adds graphic overview

Usage:

```
ISOIdentification$addGraphicOverview(graphicOverview)
```

Arguments:

`graphicOverview` object of class ISOBrowseGraphic

Returns: `TRUE` if added, `FALSE` otherwise

Method `setGraphicOverview()`**:** Sets graphic overview

Usage:

```
ISOIdentification$setGraphicOverview(graphicOverview)
```

Arguments:

`graphicOverview` object of class ISOBrowseGraphic

Returns: `TRUE` if set, `FALSE` otherwise

Method `delGraphicOverview()`**:** Deletes graphic overview

Usage:

```
ISOIdentification$delGraphicOverview(graphicOverview)
```

Arguments:

`graphicOverview` object of class ISOBrowseGraphic

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addFormat()`**:** Adds format

Usage:

```
ISOIdentification$addFormat(format)
```

Arguments:

`format` object of class ISOFormat

Returns: `TRUE` if added, `FALSE` otherwise

Method `delFormat()`**:** Deletes format

Usage:

```
ISOIdentification$delFormat(format)
```

Arguments:

`format` object of class ISOFormat

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addKeywords()`**:** Adds keywords

Usage:

```
ISOIdentification$addKeywords(keywords)
```

Arguments:

`keywords` object of class ISOKeywords

Returns: `TRUE` if added, `FALSE` otherwise

Method `setKeywords()`**:** Set keywords

Usage:

```
ISOIdentification$setKeywords(keywords)
```

Arguments:

`keywords` object of class ISOKeywords

Returns: `TRUE` if set, `FALSE` otherwise

Method `delKeywords()`**:** Deletes keywords

Usage:

```
ISOIdentification$delKeywords(keywords)
```

Arguments:

`keywords` object of class ISOKeywords

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addResourceConstraints()`**:** Adds resource constraints

Usage:

```
ISOIdentification$addResourceConstraints(resourceConstraints)
```

Arguments:

`resourceConstraints` object of class ISOConstraints

Returns: `TRUE` if added, `FALSE` otherwise

Method `setResourceConstraints()`**:** Sets resource constraints

Usage:

```
ISOIdentification$setResourceConstraints(resourceConstraints)
```

Arguments:

`resourceConstraints` object of class ISOConstraints

Returns: `TRUE` if set, `FALSE` otherwise

Method `delResourceConstraints()`**:** Deletes resource constraints

Usage:

```
ISOIdentification$delResourceConstraints(resourceConstraints)
```

Arguments:

`resourceConstraints` object of class ISOConstraints

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addAggregateInformation()`**:** Adds aggregate information

Usage:

```
ISOIdentification$addAggregateInformation(aggregateInfo)
```

Arguments:

`aggregateInfo` object of class ISOAggregateInformation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delAggregateInformation()`**:** Deletes aggregate information

Usage:

```
ISOIdentification$delAggregateInformation(aggregateInfo)
```

Arguments:

`aggregateInfo` object of class ISOAggregateInformation

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOIdentifier` *ISOIdentifier*

Description

ISOIdentifier

ISOIdentifier

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Identifier

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOIdentifier`

Public fields

`authority` authority [0..1]: ISOCitation

`code` code[1..1]: character

Methods

Public methods:

- `ISOIdentifier$new()`
- `ISOIdentifier$setAuthority()`
- `ISOIdentifier$clone()`

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

Usage:

```
ISOIdentifier$new(xml = NULL, code = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`code` code

Method `setAuthority()`**:** Set authority

Usage:

```
ISOIdentifier$setAuthority(authority)
```

Arguments:

`authority` object of class ISOCitation

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Abstract ISO class

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOImageDescription` *ISOImageDescription*

Description

ISOImageDescription

ISOImageDescription

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOImageDescription

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOContentInformation` -> `geometa::ISOCoverageDescription` -> `ISOImageDescription`

Public fields

`illuminationElevationAngle` illuminationElevationAngle [0..1]

`illuminationAzimuthAngle` illuminationAzimuthAngle [0..1]

`imagingCondition` imagingCondition [0..1]

`imageQualityCode` imageQualityCode [0..1]

`cloudCoverPercentage` cloudCoverPercentage [0..1]

`processingLevelCode` processingLevelCode [0..1]

`compressionGenerationQuantity` compressionGenerationQuantity [0..1]

`triangulationIndicator` triangulationIndicator [0..1]

`radiometricCalibrationDataAvailability` radiometricCalibrationDataAvailability [0..1]

`cameraCalibrationInformationAvailability` cameraCalibrationInformationAvailability [0..1]

`filmDistortionInformationAvailability` filmDistortionInformationAvailability [0..1]

`lensDistortionInformationAvailability` lensDistortionInformationAvailability [0..1]

Methods

Public methods:

- `ISOImageDescription$new()`
- `ISOImageDescription$setIlluminationElevationAngle()`
- `ISOImageDescription$setIlluminationAzimuthAngle()`
- `ISOImageDescription$setImagingCondition()`
- `ISOImageDescription$setImageQualityCode()`
- `ISOImageDescription$setCloudCoverPercentage()`
- `ISOImageDescription$setProcessingLevelCode()`
- `ISOImageDescription$setCompressionGenerationQuantity()`
- `ISOImageDescription$setTriangulationIndicator()`
- `ISOImageDescription$setRadiometricCalibrationDataAvailability()`
- `ISOImageDescription$setCameraCalibrationInformationAvailability()`
- `ISOImageDescription$setFilmDistortionInformationAvailability()`
- `ISOImageDescription$setLensDistortionInformationAvailability()`
- `ISOImageDescription$clone()`

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

Usage:

```
ISOImageDescription$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setIlluminationElevationAngle()`**:** Set illumination elevation angle

Usage:

```
ISOImageDescription$setIlluminationElevationAngle(illuminationElevationAngle)
```

Arguments:

`illuminationElevationAngle` object of class numeric

Method `setIlluminationAzimuthAngle()`**:** Set illumination azimuth angle

Usage:

```
ISOImageDescription$setIlluminationAzimuthAngle(illuminationAzimuthAngle)
```

Arguments:

`illuminationAzimuthAngle` object of class numeric

Method `setImagingCondition()`**:** Set imaging condition

Usage:

```
ISOImageDescription$setImagingCondition(imagingCondition)
```

Arguments:

`imagingCondition` object of class ISOImagingCondition or character among values returned by `ISOImagingCondition$values()`

Method `setImageQualityCode()`**:** Set image quality code

Usage:

```
ISOImageDescription$setImageQualityCode(code)
```

Arguments:

`code` object of class ISOMetaIdentifier

Method `setCloudCoverPercentage()`**:** Set cloud cover percentage

Usage:

```
ISOImageDescription$setCloudCoverPercentage(cloudCoverPercentage)
```

Arguments:

`cloudCoverPercentage` object of class numeric

Method `setProcessingLevelCode()`**:** Set processing level code

Usage:

```
ISOImageDescription$setProcessingLevelCode(code)
```

Arguments:

`code` object of class ISOMetaIdentifier

Method `setCompressionGenerationQuantity()`**:** Set compression generation quantity

Usage:

```
ISOImageDescription$setCompressionGenerationQuantity(quantity)
```

Arguments:

`quantity` object of class integer

Method `setTriangulationIndicator()`**:** Set triangulation indicator

Usage:

```
ISOImageDescription$setTriangulationIndicator(triangulationIndicator)
```

Arguments:

`triangulationIndicator` object of class logical

Method `setRadiometricCalibrationDataAvailability()`: Set radiometric calibration data availability

Usage:

```
ISOImageDescription$setRadiometricCalibrationDataAvailability(
  radiometricCalibrationDataAvailability
)
```

Arguments:

`radiometricCalibrationDataAvailability` object of class logical

Method `setCameraCalibrationInformationAvailability()`: Set camera calibration information availability

Usage:

```
ISOImageDescription$setCameraCalibrationInformationAvailability(
  cameraCalibrationInformationAvailability
)
```

Arguments:

`cameraCalibrationInformationAvailability` object of class logical

Method `setFilmDistortionInformationAvailability()`: Set film distortion information availability

Usage:

```
ISOImageDescription$setFilmDistortionInformationAvailability(
  filmDistortionInformationAvailability
)
```

Arguments:

`filmDistortionInformationAvailability` object of class logical

Method `setLensDistortionInformationAvailability()`: Set lens distortion information availability

Usage:

```
ISOImageDescription$setLensDistortionInformationAvailability(
  lensDistortionInformationAvailability
)
```

Arguments:

`lensDistortionInformationAvailability` object of class logical

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create image description
md <- ISOImageDescription$new()
md$setAttributeDescription("test")
md$setContentType("modelResult")

#adding 3 arbitrary dimensions
for(i in 1:3){
   band <- ISOBand$new()
 mn <- ISOMemberName$new(aName = sprintf("name %s",i), attributeType = sprintf("type %s",i))
   band$setSequenceIdentifier(mn)
   band$setDescriptor("descriptor")
   band$setMaxValue(10)
   band$setMinValue(1)
   gml <- GMLBaseUnit$new(id = sprintf("ID%s",i))
   gml$setDescriptionReference("someref")
   gml$setIdentifier("identifier", "codespace")
   gml$addName("name1", "codespace")
   gml$addName("name2", "codespace")
   gml$setQuantityTypeReference("someref")
   gml$setCatalogSymbol("symbol")
   gml$setUnitsSystem("somelink")
   band$setUnits(gml)
   band$setPeakResponse(9)
   band$setBitsPerValue(5)
   band$setToneGradation(100)
   band$setScaleFactor(1)
   band$setOffset(4)
   md$addDimension(band)
}

md$setIlluminationElevationAngle(15)
md$setIlluminationAzimuthAngle(10)
md$setImagingCondition("rain")
md$setImageQualityCode("bad")
md$setCloudCoverPercentage(90)
md$setProcessingLevelCode("high")
md$setCompressionGenerationQuantity(1L)
md$setTriangulationIndicator(FALSE)
md$setRadiometricCalibrationDataAvailability(FALSE)
md$setCameraCalibrationInformationAvailability(FALSE)
md$setFilmDistortionInformationAvailability(FALSE)
md$setLensDistortionInformationAvailability(FALSE)
```

```
xml <- md$encode()
```

`ISOImageryAbstractGeolocationInformation`
ISOImageryAbstractGeolocationInformation

Description

ISOImageryAbstractGeolocationInformation

ISOImageryAbstractGeolocationInformation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOimagery geolocation information

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryAbstractGeolocationInformation`

Methods

Public methods:

- `ISOImageryAbstractGeolocationInformation$new()`
- `ISOImageryAbstractGeolocationInformation$clone()`

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

Usage:

```
ISOImageryAbstractGeolocationInformation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

abstract class

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

---

`ISOImageryAcquisitionInformation`
ISOImageryAcquisitionInformation

Description

ISOImageryAcquisitionInformation

ISOImageryAcquisitionInformation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Imagery AcquisitionInformation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryAcquisitionInformation`

Public fields

`instrument` instrument [0..*]: ISOImageryInstrument

`operation` operation [0..*]: ISOImageryOperation

`platform` platform [0..*]: ISOImageryPlatform

`acquisitionPlan` acquisitionPlan [0..*]: ISOImageryPlan

`objective` objective [0..*]: ISOImageryObjective

`acquisitionRequirement` acquisitionRequirement [0..*]: ISOImageryRequirement

`environmentalConditions` environmentalConditions [0..1]: ISOImageryEnvironmentalRecord

Methods

Public methods:

- `ISOImageryAcquisitionInformation$new()`
- `ISOImageryAcquisitionInformation$addInstrument()`
- `ISOImageryAcquisitionInformation$delInstrument()`
- `ISOImageryAcquisitionInformation$addOperation()`
- `ISOImageryAcquisitionInformation$delOperation()`
- `ISOImageryAcquisitionInformation$addPlatform()`
- `ISOImageryAcquisitionInformation$delPlatform()`
- `ISOImageryAcquisitionInformation$addPlan()`
- `ISOImageryAcquisitionInformation$delPlan()`
- `ISOImageryAcquisitionInformation$addObjective()`
- `ISOImageryAcquisitionInformation$delObjective()`
- `ISOImageryAcquisitionInformation$addRequirement()`
- `ISOImageryAcquisitionInformation$delRequirement()`
- `ISOImageryAcquisitionInformation$setEnvironmentConditions()`
- `ISOImageryAcquisitionInformation$clone()`

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

Usage:

```
ISOImageryAcquisitionInformation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addInstrument()`**:** Adds instrument

Usage:

```
ISOImageryAcquisitionInformation$addInstrument(instrument)
```

Arguments:

`instrument` object of class ISOImageryInstrument

Returns: `TRUE` if added, `FALSE` otherwise

Method `delInstrument()`**:** Deletes instrument

Usage:

```
ISOImageryAcquisitionInformation$delInstrument(instrument)
```

Arguments:

`instrument` object of class ISOImageryInstrument

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addOperation()`**:** Adds operation

Usage:

```
ISOImageryAcquisitionInformation$addOperation(operation)
```

Arguments:

`operation` object of class ISOImageryOperation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delOperation()`: Deletes operation

Usage:

```
ISOImageryAcquisitionInformation$delOperation(operation)
```

Arguments:

`operation` object of class ISOImageryOperation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addPlatform()`: Adds platform

Usage:

```
ISOImageryAcquisitionInformation$addPlatform(platform)
```

Arguments:

`platform` object of class ISOImageryPlatform

Returns: `TRUE` if added, `FALSE` otherwise

Method `delPlatform()`: Deletes platform

Usage:

```
ISOImageryAcquisitionInformation$delPlatform(platform)
```

Arguments:

`platform` object of class ISOImageryPlatform

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addPlan()`: Adds plan

Usage:

```
ISOImageryAcquisitionInformation$addPlan(plan)
```

Arguments:

`plan` object of class ISOImageryPlan

Returns: `TRUE` if added, `FALSE` otherwise

Method `delPlan()`: Deletes plan

Usage:

```
ISOImageryAcquisitionInformation$delPlan(plan)
```

Arguments:

`plan` object of class ISOImageryPlan

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addObjective()`: Adds objective

Usage:

`ISOImageryAcquisitionInformation$addObjective(objective)`

Arguments:

`objective` object of class ISOImageryObjective

Returns: `TRUE` if added, `FALSE` otherwise

Method `delObjective()`**:** Deletes objective

Usage:

`ISOImageryAcquisitionInformation$delObjective(objective)`

Arguments:

`objective` object of class ISOImageryObjective

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addRequirement()`**:** Adds requirement

Usage:

`ISOImageryAcquisitionInformation$addRequirement(requirement)`

Arguments:

`requirement` object of class ISOImageryRequirement

Returns: `TRUE` if added, `FALSE` otherwise

Method `delRequirement()`**:** Deletes requirement

Usage:

`ISOImageryAcquisitionInformation$delRequirement(requirement)`

Arguments:

`requirement` object of class ISOImageryRequirement

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setEnvironmentConditions()`**:** Set environment conditions

Usage:

`ISOImageryAcquisitionInformation$setEnvironmentConditions(conditions)`

Arguments:

`conditions` object of class ISOImageryEnvironmentalRecord

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

Usage:

`ISOImageryAcquisitionInformation$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – AcquisitionInformation – Part 2: Extensions for imagery and gridded data

Examples

```
 md = ISOImageryAcquisitionInformation$new()

xml <- md$encode()
```

ISOImageryAlgorithm *ISOImageryAlgorithm*

Description

ISOImageryAlgorithm

ISOImageryAlgorithm

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery algorithm

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryAlgorithm`

Public fields

`citation` citation [1..1]: ISOCitation

`description` description [1..1]: character|ISOLocalisedCharacterString

Methods

Public methods:

- `ISOImageryAlgorithm$new()`
- `ISOImageryAlgorithm$setCitation()`
- `ISOImageryAlgorithm$setDescription()`
- `ISOImageryAlgorithm$clone()`

Method `new()`**:** Initialized object

Usage:

```
ISOImageryAlgorithm$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setCitation():` Set citation

Usage:

```
ISOImageryAlgorithm$setCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Method `setDescription():` Set description

Usage:

```
ISOImageryAlgorithm$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` list of localized texts. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryAlgorithm$new()

#add citation
rp1 <- ISOResponsibleParty$new()
rp1$setIndividualName("someone1")
rp1$setOrganisationName("somewhere1")
rp1$setPositionName("someposition1")
rp1$setRole("pointOfContact")
contact1 <- ISOContact$new()
phone1 <- ISOTelephone$new()
phone1$setVoice("myphonenumber1")
phone1$setFacsimile("myfacsimile1")
contact1$setPhone(phone1)
```

```
address1 <- ISOAddress$new()
address1$setDeliveryPoint("theaddress1")
address1$setCity("thecity1")
address1$setPostalCode("111")
address1$setCountry("France")
address1$setEmail("someone1@theorg.org")
contact1$setAddress(address1)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact1$setOnlineResource(res)
rp1$setContactInfo(contact1)

#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(ISOdate(2015,1,1))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp1)
md$setCitation(ct)
md$setDescription("some description")

xml <- md$encode()
```

ISOImageryBand *ISOImageryBand*

Description

ISOImageryBand

ISOImageryBand

Format

R6Class object.

Value

Object of R6Class for modelling an ISO imagery band

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISORangeDimension -> geometa::ISOBand -> ISOImageryBand

Public fields

`bandBoundaryDefinition` bandBoundaryDefinition [0..1]: ISOImageryBandDefinition

`nominalSpatialResolution` nominalSpatialResolution [0..1] ISOBaseReal

`transferFunctionType` transferFunctionType [0..1]: ISOImageryTransferFunctionType

`transmittedPolarisation` transmittedPolarisation [0..1]: ISOImageryPolarisationOrientation

`detectedPolarisation` detectedPolarisation [0..1]: ISOImageryPolarisationOrientation

Methods

Public methods:

- `ISOImageryBand$new()`
- `ISOImageryBand$setBandBoundaryDefinition()`
- `ISOImageryBand$setNominalSpatialResolution()`
- `ISOImageryBand$setTransferFunctionType()`
- `ISOImageryBand$setTransmittedPolarisation()`
- `ISOImageryBand$setDetectedPolarisation()`
- `ISOImageryBand$clone()`

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

Usage:

```
ISOImageryBand$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setBandBoundaryDefinition()`**:** Set band boundary definition

Usage:

```
ISOImageryBand$setBandBoundaryDefinition(definition)
```

Arguments:

`definition` object of class ISOImageryBandDefinition or character among values returned by `ISOImageryBandDefinition$values()`

Method `setNominalSpatialResolution()`**:** Set nominal spatial resolution

Usage:

```
ISOImageryBand$setNominalSpatialResolution(resolution)
```

Arguments:

`resolution` object of class numeric

Method `setTransferFunctionType()`**:** Set transfer function type

Usage:

```
ISOImageryBand$setTransferFunctionType(functionType)
```

Arguments:

`functionType` object of class ISOImageryTransferFunctionType or any character from values returned by `ISOImageryTransferFunctionType$values()`

Method `setTransmittedPolarisation()`**:** Set transmitted polarisation

Usage:

```
ISOImageryBand$setTransmittedPolarisation(polarisation)
```

Arguments:

`polarisation` object of class ISOImageryPolarisationOrientation or any character from values returned by `ISOImageryPolarisationOrientation$values()`

Method `setDetectedPolarisation()`**:** Set detected polarisation

Usage:

```
ISOImageryBand$setDetectedPolarisation(polarisation)
```

Arguments:

`polarisation` object of class ISOImageryPolarisationOrientation or any character from values returned by `ISOImageryPolarisationOrientation$values()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```
#create band range dimension
md <- ISOImageryBand$new()
md$setSequenceIdentifier(ISOMemberName$new(aName = "name", attributeType = "type"))
md$setDescriptor("descriptor")
md$setMaxValue(10)
md$setMinValue(1)
gml <- GMLBaseUnit$new(id = "ID")
gml$setDescriptionReference("someref")
gml$setIdentifier("identifier", "codespace")
gml$addName("name1", "codespace")
gml$addName("name2", "codespace")
gml$setQuantityTypeReference("someref")
gml$setCatalogSymbol("symbol")
gml$setUnitsSystem("somelink")
md$setUnits(gml)
md$setPeakResponse(9)
md$setBitsPerValue(5)
md$setToneGradation(100)
md$setScaleFactor(1)
md$setOffset(4)

md$setBandBoundaryDefinition("fiftyPercent")
```

```
md$setNominalSpatialResolution(14.5)
md$setTransferFunctionType("linear")
md$setTransmittedPolarisation("horizontal")
md$setDetectedPolarisation("horizontal")

xml <- md$encode()
```

`ISOImageryBandDefinition` *ISOImageryBandDefinition*

Description

ISOImageryBandDefinition

ISOImageryBandDefinition

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Imagery Band definition

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOImageryBandDefinition`

Methods

Public methods:

- `ISOImageryBandDefinition$new()`
- `ISOImageryBandDefinition$clone()`

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

Usage:

`ISOImageryBandDefinition$new(xml = NULL, value, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOImageryBandDefinition$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#possible values
values <- ISOImageryBandDefinition$values(labels = TRUE)

#some def
fiftyp <- ISOImageryBandDefinition$new(value = "fiftyPercent")
```

`ISOImageryContext` *ISOImageryContext*

Description

ISOImageryContext

ISOImageryContext

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Imagery Context

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOImageryContext`

Methods

Public methods:

- `ISOImageryContext$new()`
- `ISOImageryContext$clone()`

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

Usage:

```
ISOImageryContext$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#possible values
values <- ISOImageryContext$values(labels = TRUE)

#some def
acquisition <- ISOImageryContext$new(value = "acquisition")
```

ISOImageryCoverageDescription
ISOImageryCoverageDescription

Description

ISOImageryCoverageDescription

ISOImageryCoverageDescription

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery image description

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOContentInformation` -> `geometa::ISOCoverageDescription` -> `ISOImageryCoverageDescription`

Public fields

`rangeElementDescription` rangeElementDescription [0..*] : ISOImageryRangeElementDescription

Methods

Public methods:

- `ISOImageryCoverageDescription$new()`
- `ISOImageryCoverageDescription$addRangeElementDescription()`
- `ISOImageryCoverageDescription$delRangeElementDescription()`
- `ISOImageryCoverageDescription$clone()`

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

Usage:

```
ISOImageryCoverageDescription$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addRangeElementDescription()`**:** Adds range element description

Usage:

```
ISOImageryCoverageDescription$addRangeElementDescription(description)
```

Arguments:

`description` object of class ISOImageryRangeElementDescription

Returns: `TRUE` if added, `FALSE` otherwise

Method `delRangeElementDescription()`**:** Deletes range element description

Usage:

```
ISOImageryCoverageDescription$delRangeElementDescription(description)
```

Arguments:

`description` object of class ISOImageryRangeElementDescription

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#create coverage description
md <- ISOImageryCoverageDescription$new()
md$setAttributeDescription("test")
md$setContentType("modelResult")

#adding 3 arbitrary dimensions
for(i in 1:3){
   band <- ISOBand$new()
 mn <- ISOMemberName$new(aName = sprintf("name %s",i), attributeType = sprintf("type %s",i))
   band$setSequenceIdentifier(mn)
   band$setDescriptor("descriptor")
   band$setMaxValue(10)
   band$setMinValue(1)
   gml <- GMLBaseUnit$new(id = sprintf("ID%s",i))
   gml$setDescriptionReference("someref")
   gml$setIdentifier("identifier", "codespace")
   gml$addName("name1", "codespace")
   gml$addName("name2", "codespace")
   gml$setQuantityTypeReference("someref")
   gml$setCatalogSymbol("symbol")
   gml$setUnitsSystem("somelink")
   band$setUnits(gml)
   band$setPeakResponse(9)
   band$setBitsPerValue(5)
   band$setToneGradation(100)
   band$setScaleFactor(1)
   band$setOffset(4)
   md$addDimension(band)
}

des <- ISOImageryRangeElementDescription$new()
des$setName("name")
des$setDefinition("description")
des$addRangeElement("record1")
des$addRangeElement("record2")
md$addRangeElementDescription(des)
xml <- md$encode()
```

ISOImageryCoverageResult *ISOImageryCoverageResult*

Description

ISOImageryCoverageResult

ISOImageryCoverageResult

Format

R6Class object.

Value

Object of R6Class for modelling an ISO imagery coverage result

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISOAbstractResult -> ISOImageryCoverageResult

Public fields

`spatialRepresentationType` spatialRepresentationType [1..1] : ISOSpatialRepresentationType

`resultFile` resultFile [1..1]: ISODataFile

`resultSpatialRepresentation` resultSpatialRepresentation [1..1]: ISOSpatialRepresentation

`resultContentDescription` resultContentDescription [1..1]: ISOCoverageDescription

`resultFormat` resultFormat [1..1]: ISOFormat

Methods

Public methods:

- `ISOImageryCoverageResult$new()`
- `ISOImageryCoverageResult$setSpatialRepresentationType()`
- `ISOImageryCoverageResult$setResultFile()`
- `ISOImageryCoverageResult$setResultSpatialRepresentation()`
- `ISOImageryCoverageResult$setResultCoverageDescription()`
- `ISOImageryCoverageResult$setResultFormat()`
- `ISOImageryCoverageResult$clone()`

Method `new()`: Initializes object

Usage:

```
ISOImageryCoverageResult$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setSpatialRepresentationType():` Set spatial representation type

Usage:

```
ISOImageryCoverageResult$setSpatialRepresentationType(
  spatialRepresentationType
)
```

Arguments:

`spatialRepresentationType` object of class ISOSpatialRepresentationType or character among values returned by `ISOSpatialRepresentationType$values()`

Method `setResultFile():` Set result file

Usage:

```
ISOImageryCoverageResult$setResultFile(resultFile)
```

Arguments:

`resultFile` object of class ISODataFile

Method `setResultSpatialRepresentation():` Set result spatial representation

Usage:

```
ISOImageryCoverageResult$setResultSpatialRepresentation(spatialRepresentation)
```

Arguments:

`spatialRepresentation` object of class ISOSpatialRepresentation

Method `setResultCoverageDescription():` Set result coverage description

Usage:

```
ISOImageryCoverageResult$setResultCoverageDescription(coverageDescription)
```

Arguments:

`coverageDescription` object of class ISOCoverageDescription

Method `setResultFormat():` Set format

Usage:

```
ISOImageryCoverageResult$setResultFormat(format)
```

Arguments:

`format` object of class ISOFormat

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

---

`ISOImageryEnvironmentalRecord` *ISOImageryEnvironmentalRecord*

Description

ISOImageryEnvironmentalRecord

ISOImageryEnvironmentalRecord

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery environmental record

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryEnvironmentalRecord`

Public fields

`averageAirTemperature` averageAirTemperature

`maxRelativeHumidity` maxRelativeHumidity

`maxAltitude` maxAltitude

`meterologicalConditions` meterologicalConditions

Methods

Public methods:

- `ISOImageryEnvironmentalRecord$new()`
- `ISOImageryEnvironmentalRecord$setAverageAirTemperature()`
- `ISOImageryEnvironmentalRecord$setMaxRelativeHumidity()`
- `ISOImageryEnvironmentalRecord$setMaxAltitude()`
- `ISOImageryEnvironmentalRecord$setMeterologicalConditions()`
- `ISOImageryEnvironmentalRecord$clone()`

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

Usage:

```
ISOImageryEnvironmentalRecord$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setAverageAirTemperature()`**:** Set average air temperature

Usage:

```
ISOImageryEnvironmentalRecord$setAverageAirTemperature(temperature)
```

Arguments:

`temperature` object of class numeric

Method `setMaxRelativeHumidity()`**:** Set max relative humidity

Usage:

```
ISOImageryEnvironmentalRecord$setMaxRelativeHumidity(humidity)
```

Arguments:

`humidity` object of class numeric

Method `setMaxAltitude()`**:** Set max altitude

Usage:

```
ISOImageryEnvironmentalRecord$setMaxAltitude(altitude)
```

Arguments:

`altitude` object of class numeric

Method `setMeterologicalConditions()`**:** Set meterological conditions

Usage:

```
ISOImageryEnvironmentalRecord$setMeterologicalConditions(
  conditions,
  locales = NULL
)
```

Arguments:

`conditions` conditions

`locales` list of localized texts. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryEnvironmentalRecord$new()
md$setAverageAirTemperature(3)
md$setMaxRelativeHumidity(67)
md$setMaxAltitude(400)
md$setMeterologicalConditions("some conditions")
xml <- md$encode()
```

`ISOImageryEvent` *ISOImageryEvent*

Description

ISOImageryEvent

ISOImageryEvent

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery event

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryEvent`

Public fields

`identifier` identifier [1..1]: ISOMetaIdentifier

`trigger` trigger [1..1]: ISOImageryTrigger

`context` context [1..1]: ISOImageryContext

`sequence` sequence [1..1]: ISOImagerySequence

`time` time [1..1]: POSIXt

`relatedPass` relatedPass [0..1]: ISOImageryPlatformPass

`relatedSensor` relatedSensor [0..*]: ISOImageryInstrument

`expectedObjective` expectedObjective [0..*]: ISOImageryObjective

Methods

Public methods:

- `ISOImageryEvent$new()`
- `ISOImageryEvent$setIdentifier()`
- `ISOImageryEvent$setTrigger()`
- `ISOImageryEvent$setContext()`
- `ISOImageryEvent$setSequence()`
- `ISOImageryEvent$setTime()`
- `ISOImageryEvent$setPlatformPass()`
- `ISOImageryEvent$addSensor()`
- `ISOImageryEvent$delSensor()`
- `ISOImageryEvent$addObjective()`
- `ISOImageryEvent$delObjective()`
- `ISOImageryEvent$clone()`

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

Usage:

```
ISOImageryEvent$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setIdentifier()`**:** Set identifier

Usage:

```
ISOImageryEvent$setIdentifier(identifier)
```

Arguments:

`identifier` object of class ISOMetaIdentifier or character

Method `setTrigger()`**:** Set trigger

Usage:

```
ISOImageryEvent$setTrigger(trigger)
```

Arguments:

`trigger` object of class ISOImageryTrigger or any character among values returned by `ISOImageryTrigger$values()`

Method `setContext()`**:** Set context

Usage:

```
ISOImageryEvent$setContext(context)
```

Arguments:

`context` object of class ISOImageryContext or any character among values returned by `ISOImageryContext$values()`

Method `setSequence()`**:** Set sequence

Usage:

```
ISOImageryEvent$setSequence(sequence)
```

Arguments:

`sequence` object of class ISOImagerySequence or any character among values returned by `ISOImagerySequence$values()`

Method `setTime():` Set time

Usage:

```
ISOImageryEvent$setTime(time)
```

Arguments:

`time` object of class POSIXct

Method `setPlatformPass():` Set platform pass

Usage:

```
ISOImageryEvent$setPlatformPass(platformPass)
```

Arguments:

`platformPass` object of class ISOImageryPlatformPass

Method `addSensor():` Adds sensor

Usage:

```
ISOImageryEvent$addSensor(sensor)
```

Arguments:

`sensor` object of class ISOImageryInstrument

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSensor():` Deletes sensor

Usage:

```
ISOImageryEvent$delSensor(sensor)
```

Arguments:

`sensor` object of class ISOImageryInstrument

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addObjective():` Adds objective

Usage:

```
ISOImageryEvent$addObjective(objective)
```

Arguments:

`objective` object of class ISOImageryObjective

Returns: `TRUE` if added, `FALSE` otherwise

Method `delObjective():` Deletes objective

Usage:

```
ISOImageryEvent$delObjective(objective)
```

Arguments:

`objective` object of class ISOImageryObjective

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOImageryEvent$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryEvent$new()
md$setIdentifier("event_1")
md$setTrigger("manual")
md$setContext("pass")
md$setSequence("instantaneous")
md$setTime(Sys.time())

xml <- md$encode()
```

`ISOImageryGCP` *ISOImageryGCPCollection*

Description

ISOImageryGCPCollection

ISOImageryGCPCollection

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery gcp collection

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLAbstractObject` -> `ISOImageryGCP`

Public fields

`geographicCoordinates` geographicCoordinates

Methods

Public methods:

- `ISOImageryGCP$new()`
- `ISOImageryGCP$setGeographicCoordinates()`
- `ISOImageryGCP$clone()`

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

Usage:

`ISOImageryGCP$new(xml = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

Method `setGeographicCoordinates()`**:** Set geographic coordinates

Usage:

`ISOImageryGCP$setGeographicCoordinates(sfg = NULL, m = NULL)`

Arguments:

`sfg` simple feature object from **sf**

`m` object of class matrix

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

Usage:

`ISOImageryGCP$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryGCP$new()
require(sf)
pt <- sf::st_point(c(1,1))
md$setGeographicCoordinates(sfg = pt)
xml <- md$encode()
```

`ISOImageryGCPCollection` *ISOImageryGCPCollection*

Description

ISOImageryGCPCollection

ISOImageryGCPCollection

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery gcp collection

Methods

`new(xml)` This method is used to instantiate an `ISOImageryGCPCollection`

`setCollectionIdentification(id)` Set the identifier, object of class `integer`

`setCollectionName(name, locales)` Sets a name (object of class "character"). Locale names can be specified as `list` with the `locales` argument.

`setCoordinateReferenceSystem(crs)` Sets the crs, object of class `ISOReferenceSystem`

`addGCP(gcp)` Adds a GCP, object of class `ISOImageryGCP`

`delGCP(gcp)` Deletes a GCP, object of class `ISOImageryGCP`

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOImageryAbstractGeolocationInformat`
-> `ISOImageryGCPCollection`

Public fields

`collectionIdentification` collectionIdentification [1..1]: integer

`collectionName` collectionName [1..1]: character|ISOLocalisedCharacterString

`coordinateReferenceSystem` coordinateReferenceSystem [1..1]: ISOReferenceSystem

`gcp` gcp [0..*]: list of ISOImageryGCP

Methods

Public methods:

- `ISOImageryGCPCollection$new()`
- `ISOImageryGCPCollection$setCollectionIdentification()`
- `ISOImageryGCPCollection$setCollectionName()`
- `ISOImageryGCPCollection$setCoordinateReferenceSystem()`
- `ISOImageryGCPCollection$addGCP()`
- `ISOImageryGCPCollection$delGCP()`
- `ISOImageryGCPCollection$clone()`

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

Usage:

```
ISOImageryGCPCollection$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setCollectionIdentification()`**:** Set collection identification id

Usage:

```
ISOImageryGCPCollection$setCollectionIdentification(id)
```

Arguments:

`id` object of class integer

Method `setCollectionName()`**:** Set collection name

Usage:

```
ISOImageryGCPCollection$setCollectionName(name, locales = NULL)
```

Arguments:

`name` object of class character

`locales` list of localized names. Default is `NULL`

Method `setCoordinateReferenceSystem()`**:** Set coordinate reference system

Usage:

```
ISOImageryGCPCollection$setCoordinateReferenceSystem(crs)
```

Arguments:

`crs` object of class ISOReferenceSystem

Method `addGCP()`**:** Adds GCP

Usage:

```
ISOImageryGCPCollection$addGCP(gcp)
```

Arguments:

`gcp` object of class ISOImageryGCP

Returns: `TRUE` if added, `FALSE` otherwise

Method `delGCP()`**:** Deletes GCP

Usage:

`ISOImageryGCPCollection$delGCP(gcp)`

Arguments:

`gcp` object of class ISOImageryGCP

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOImageryGCPCollection$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryGCPCollection$new()
md$setCollectionIdentification(1L)
md$setCollectionName("name")
rs <- ISOReferenceSystem$new()
rsId <- ISOReferenceIdentifier$new(code = "4326", codeSpace = "EPSG")
rs$setReferenceSystemIdentifier(rsId)
md$setCoordinateReferenceSystem(rs)
xml <- md$encode()
```

`ISOImageryGeometryType` *ISOImageryGeometryType*

Description

ISOImageryGeometryType

ISOImageryGeometryType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Imagery geometry type

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOImageryGeometryType`

Methods

Public methods:

- `ISOImageryGeometryType$new()`
- `ISOImageryGeometryType$clone()`

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

Usage:

```
ISOImageryGeometryType$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#possible values
values <- ISOImageryGeometryType$values(labels = TRUE)

#some def
point <- ISOImageryGeometryType$new(value = "point")
```

`ISOImageryGeorectified` *ISOImageryGeorectified*

Description

ISOImageryGeorectified

ISOImageryGeorectified

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO image Georectified

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOSpatialRepresentation` -> `geometa::ISOGridSpatialRepresentation` -> `geometa::ISOGeorectified` -> `ISOImageryGeorectified`

Public fields

`checkPoint` checkPoint [0..*]: ISOImageryGCP

Methods

Public methods:

- `ISOImageryGeorectified$new()`
- `ISOImageryGeorectified$addCheckPoint()`
- `ISOImageryGeorectified$delCheckPoint()`
- `ISOImageryGeorectified$clone()`

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

Usage:

```
ISOImageryGeorectified$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addCheckPoint()`**:** Adds check point

Usage:

```
ISOImageryGeorectified$addCheckPoint(sfg = NULL, m = NULL)
```

Arguments:

`sfg` simple feature object from **sf**

`m` object of class matrix

Returns: `TRUE` if added, `FALSE` otherwise

Method `delCheckPoint()`: Deletes check point

Usage:

```
ISOImageryGeorectified$delCheckPoint(sfg = NULL, m = NULL)
```

Arguments:

`sfg` simple feature object from **sf**

`m` object of class matrix

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata – Part 2: Extensions for imagery and gridded data

---

`ISOImageryGeoreferenceable` *ISOImageryGeoreferenceable*

Description

ISOImageryGeoreferenceable

ISOImageryGeoreferenceable

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery Georeferenceable

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOSpatialRepresentation` -> `geometa::ISOGridSpatialRepresentation` -> `geometa::ISOGeoreferenceable` -> `ISOImageryGeoreferenceable`

Public fields

`geolocationInformation` geolocationInformation [0..*]: ISOImageryGeolocationInformation

Methods

Public methods:

- `ISOImageryGeoreferenceable$new()`
- `ISOImageryGeoreferenceable$addGeolocationInformation()`
- `ISOImageryGeoreferenceable$delGeolocationInformation()`
- `ISOImageryGeoreferenceable$clone()`

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

Usage:

```
ISOImageryGeoreferenceable$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addGeolocationInformation()`**:** Adds geolocation information

Usage:

```
ISOImageryGeoreferenceable$addGeolocationInformation(geolocationInfo)
```

Arguments:

`geolocationInfo` object of class inheriting ISOImageryAbstractGeolocationInformation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delGeolocationInformation()`**:** Deletes geolocation information

Usage:

```
ISOImageryGeoreferenceable$delGeolocationInformation(geolocationInfo)
```

Arguments:

`geolocationInfo` object of class inheriting ISOImageryAbstractGeolocationInformation

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata – Part 2: Extensions for imagery and gridded data

---

`ISOImageryImageDescription` *ISOImageryImageDescription*

Description

ISOImageryImageDescription

ISOImageryImageDescription

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery image description

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOContentInformation` -> `geometa::ISOCoverageDescription` -> `geometa::ISOImageDescription` -> `ISOImageryImageDescription`

Public fields

`rangeElementDescription` rangeElementDescription [0..*] : ISOImageryRangeElementDescription

Methods

Public methods:

- `ISOImageryImageDescription$new()`
- `ISOImageryImageDescription$addRangeElementDescription()`
- `ISOImageryImageDescription$delRangeElementDescription()`
- `ISOImageryImageDescription$clone()`

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

Usage:

```
ISOImageryImageDescription$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addRangeElementDescription()`**:** Adds range element description

Usage:

```
ISOImageryImageDescription$addRangeElementDescription(description)
```

Arguments:

`description` object of class ISOImageryRangeElementDescription

Returns: `TRUE` if added, `FALSE` otherwise

Method `delRangeElementDescription()`**:** Deletes range element description

Usage:

```
ISOImageryImageDescription$delRangeElementDescription(description)
```

Arguments:

`description` object of class ISOImageryRangeElementDescription

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#create image description
md <- ISOImageryImageDescription$new()
md$setAttributeDescription("test")
md$setContentType("modelResult")

#adding 3 arbitrary dimensions
for(i in 1:3){
   band <- ISOBand$new()
 mn <- ISOMemberName$new(aName = sprintf("name %s",i), attributeType = sprintf("type %s",i))
   band$setSequenceIdentifier(mn)
   band$setDescriptor("descriptor")
   band$setMaxValue(10)
   band$setMinValue(1)
```

```
    gml <- GMLBaseUnit$new(id = sprintf("ID%s",i))
    gml$setDescriptionReference("someref")
    gml$setIdentifier("identifier", "codespace")
    gml$addName("name1", "codespace")
    gml$addName("name2", "codespace")
    gml$setQuantityTypeReference("someref")
    gml$setCatalogSymbol("symbol")
    gml$setUnitsSystem("somelink")
    band$setUnits(gml)
    band$setPeakResponse(9)
    band$setBitsPerValue(5)
    band$setToneGradation(100)
    band$setScaleFactor(1)
    band$setOffset(4)
    md$addDimension(band)
}

md$setIlluminationElevationAngle(15)
md$setIlluminationAzimuthAngle(10)
md$setImagingCondition("rain")
md$setImageQualityCode("bad")
md$setCloudCoverPercentage(90)
md$setProcessingLevelCode("high")
md$setCompressionGenerationQuantity(1L)
md$setTriangulationIndicator(FALSE)
md$setRadiometricCalibrationDataAvailability(FALSE)
md$setCameraCalibrationInformationAvailability(FALSE)
md$setFilmDistortionInformationAvailability(FALSE)
md$setLensDistortionInformationAvailability(FALSE)

des <- ISOImageryRangeElementDescription$new()
des$setName("name")
des$setDefinition("description")
des$addRangeElement("record1")
des$addRangeElement("record2")
md$addRangeElementDescription(des)
xml <- md$encode()
```

`ISOImageryInstrument` *ISOImageryPlatform*

Description

ISOImageryPlatform

ISOImageryPlatform

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery platform

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryInstrument`

Public fields

`citation` citation [0..*]: ISOCitation

`identifier` identifier [1..1]: ISOMetaIdentifier

`type` type [1..1]: character|ISOLocalisedCharacterString

`description` description [0..1]: character|ISOLocalisedCharacterString

`mountedOn` mountedOn [0..*]: ISOImageryPlatform

Methods

Public methods:

- `ISOImageryInstrument$new()`
- `ISOImageryInstrument$addCitation()`
- `ISOImageryInstrument$delCitation()`
- `ISOImageryInstrument$setIdentifier()`
- `ISOImageryInstrument$setType()`
- `ISOImageryInstrument$setDescription()`
- `ISOImageryInstrument$addPlatform()`
- `ISOImageryInstrument$delPlatform()`
- `ISOImageryInstrument$clone()`

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

Usage:

```
ISOImageryInstrument$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addCitation()`**:** Adds citation

Usage:

```
ISOImageryInstrument$addCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delCitation()`**:** Deletes citation

Usage:

```
ISOImageryInstrument$delCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setIdentifier()`**:** Set identifier

Usage:

```
ISOImageryInstrument$setIdentifier(identifier)
```

Arguments:

`identifier` object of class ISOMetaIdentifier or character

Method `setType()`**:** Set type

Usage:

```
ISOImageryInstrument$setType(type, locales = NULL)
```

Arguments:

`type` type

`locales` list of localized texts. Default is `NULL`

Method `setDescription()`**:** Set description

Usage:

```
ISOImageryInstrument$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` list of localized texts. Default is `NULL`

Method `addPlatform()`**:** Adds platform

Usage:

```
ISOImageryInstrument$addPlatform(platform)
```

Arguments:

`platform` object of class ISOImageryPlatform

Returns: `TRUE` if added, `FALSE` otherwise

Method `delPlatform()`**:** Deletes platform

Usage:

```
ISOImageryInstrument$delPlatform(platform)
```

Arguments:

`platform` object of class ISOImageryPlatform

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryInstrument$new()
md$setIdentifier("identifier")
md$setType("type")
md$setDescription("description")
xml <- md$encode()
```

`ISOImageryMetadata` *ISOImageryMetadata*

Description

ISOImageryMetadata

ISOImageryMetadata

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Imagery Metadata

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOMetadata` -> `ISOImageryMetadata`

Public fields

`acquisitionInformation` acquisitionInformation [0..*]: ISOImageryAcquisitionInformation

Methods

Public methods:

- `ISOImageryMetadata$new()`
- `ISOImageryMetadata$addAcquisitionInfo()`
- `ISOImageryMetadata$delAcquisitionInfo()`
- `ISOImageryMetadata$clone()`

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

Usage:

```
ISOImageryMetadata$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addAcquisitionInfo()`**:** Adds acquisition info

Usage:

```
ISOImageryMetadata$addAcquisitionInfo(acquisitionInfo)
```

Arguments:

`acquisitionInfo` object of class ISOImageryAcquisitionInformation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delAcquisitionInfo()`**:** Deletes acquisition info

Usage:

```
ISOImageryMetadata$delAcquisitionInfo(acquisitionInfo)
```

Arguments:

`acquisitionInfo` object of class ISOImageryAcquisitionInformation

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata – Part 2: Extensions for imagery and gridded data

Examples

```
 #example 1 - WRITE: Create an ISO metadata and encode it as XML
 #######################################################
 md = ISOImageryMetadata$new()
 md$setFileIdentifier("my-metadata-identifier")
 md$setParentIdentifier("my-parent-metadata-identifier")
 md$setCharacterSet("utf8")
 md$setLanguage("eng")
 md$setDateStamp(ISOdate(2015, 1, 1, 1))
 md$setMetadataStandardName("ISO 19115:2003/19139")
 md$setMetadataStandardVersion("1.0")
 md$setDataSetURI("my-dataset-identifier")

 #add 3 contacts
 for(i in 1:3){
   rp <- ISOResponsibleParty$new()
   rp$setIndividualName(paste0("someone",i))
   rp$setOrganisationName("somewhere")
   rp$setPositionName(paste0("someposition",i))
   rp$setRole("pointOfContact")
   contact <- ISOContact$new()
   phone <- ISOTelephone$new()
   phone$setVoice(paste0("myphonenumber",i))
   phone$setFacsimile(paste0("myfacsimile",i))
   contact$setPhone(phone)
   address <- ISOAddress$new()
   address$setDeliveryPoint("theaddress")
   address$setCity("thecity")
   address$setPostalCode("111")
   address$setCountry("France")
   address$setEmail("someone@theorg.org")
   contact$setAddress(address)
   res <- ISOOnlineResource$new()
   res$setLinkage("http://somelink")
   res$setName("someresourcename")
   contact$setOnlineResource(res)
   rp$setContactInfo(contact)
   md$addContact(rp)
}

#VectorSpatialRepresentation
vsr <- ISOVectorSpatialRepresentation$new()
vsr$setTopologyLevel("geometryOnly")
geomObject <- ISOGeometricObjects$new()
geomObject$setGeometricObjectType("surface")
geomObject$setGeometricObjectCount(5L)
vsr$addGeometricObjects(geomObject)
md$addSpatialRepresentationInfo(vsr)

#ReferenceSystem
rs <- ISOReferenceSystem$new()
rsId <- ISOReferenceIdentifier$new(code = "4326", codeSpace = "EPSG")
```

```
rs$setReferenceSystemIdentifier(rsId)
md$addReferenceSystemInfo(rs)

#data identification
ident <- ISODataIdentification$new()
ident$setAbstract("abstract")
ident$setPurpose("purpose")
ident$addCredit("credit1")
ident$addCredit("credit2")
ident$addCredit("credit3")
ident$addStatus("completed")
ident$addLanguage("eng")
ident$addCharacterSet("utf8")
ident$addTopicCategory("biota")
ident$addTopicCategory("oceans")

#adding a point of contact
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
rp$setOrganisationName("somewhere")
rp$setPositionName("someposition")
rp$setRole("pointOfContact")
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://somelink")
res$setName("somename")
contact$setOnlineResource(res)
rp$setContactInfo(contact)
ident$addPointOfContact(rp)

#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(as.Date(ISOdate(2015, 1, 1, 1)))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp)
```

```
ident$setCitation(ct)

#graphic overview
go1 <- ISOBrowseGraphic$new(
  fileName = "http://wwww.somefile.org/png1",
  fileDescription = "Map Overview 1",
  fileType = "image/png"
)
go2 <- ISOBrowseGraphic$new(
  fileName = "http://www.somefile.org/png2",
  fileDescription = "Map Overview 2",
  fileType = "image/png"
)
ident$addGraphicOverview(go1)
ident$addGraphicOverview(go2)

#maintenance information
mi <- ISOMaintenanceInformation$new()
mi$setMaintenanceFrequency("daily")
ident$addResourceMaintenance(mi)

#adding legal constraints
lc <- ISOLegalConstraints$new()
lc$addUseLimitation("limitation1")
lc$addUseLimitation("limitation2")
lc$addUseLimitation("limitation3")
lc$addAccessConstraint("copyright")
lc$addAccessConstraint("license")
lc$addUseConstraint("copyright")
lc$addUseConstraint("license")
ident$addResourceConstraints(lc)

#adding security constraints
sc <- ISOSecurityConstraints$new()
sc$setClassification("secret")
sc$setUserNote("ultra secret")
sc$setClassificationSystem("no classification in particular")
sc$setHandlingDescription("description")
ident$addResourceConstraints(sc)

#adding extent
extent <- ISOExtent$new()
bbox <- ISOGeographicBoundingBox$new(minx = -180, miny = -90, maxx = 180, maxy = 90)
extent$addGeographicElement(bbox)
ident$addExtent(extent)

#add keywords
kwds <- ISOKeywords$new()
kwds$addKeyword("keyword1")
kwds$addKeyword("keyword2")
kwds$setKeywordType("theme")
th <- ISOCitation$new()
th$setTitle("General")
```

```
th$addDate(d)
kwds$setThesaurusName(th)
ident$addKeywords(kwds)

#add an INSPIRE spatial data theme?
inspire_kwd <- ISOKeywords$new()
anc1 <- ISOAnchor$new(
  name = "Environmental monitoring facilities",
  href = "http://inspire.ec.europa.eu/theme/ef"
)
inspire_kwd$addKeyword(anc1)
inspire_kwd$setKeywordType("theme")
th <- ISOCitation$new()
th$setTitle(
  ISOAnchor$new(
    name = "GEMET - INSPIRE themes, version 1.0",
    href="http://www.eionet.europa.eu/gemet/inspire_themes"
  )
)
inspire_date <- ISODate$new()
inspire_date$setDate(as.Date("2008-06-01"))
inspire_date$setDateType("publication")
th$addDate(inspire_date)
inspire_kwd$setThesaurusName(th)
ident$addKeywords(inspire_kwd)

#supplementalInformation
ident$setSupplementalInformation("some additional information")

#spatial representation type
ident$addSpatialRepresentationType("vector")

md$addIdentificationInfo(ident)

#Distribution
distrib <- ISODistribution$new()
dto <- ISODigitalTransferOptions$new()
for(i in 1:3){
  or <- ISOOnlineResource$new()
  or$setLinkage(paste0("http://somelink",i))
  or$setName(paste0("name",i))
  or$setDescription(paste0("description",i))
  or$setProtocol("WWW:LINK-1.0-http--link")
  dto$addOnlineResource(or)
}
distrib$addDigitalTransferOptions(dto)
md$setDistributionInfo(distrib)

#create dataQuality object with a 'dataset' scope
dq <- ISODataQuality$new()
scope <- ISOScope$new()
scope$setLevel("dataset")
dq$setScope(scope)
```

```
#add data quality reports...

#add a report the data quality
dc <- ISODomainConsistency$new()
result <- ISOConformanceResult$new()
spec <- ISOCitation$new()
spec$setTitle("Data Quality check")
spec$addAlternateTitle("This is is some data quality check report")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dc$addResult(result)
dq$addReport(dc)

#add INSPIRE reports?
#INSPIRE - interoperability of spatial data sets and services
dc_inspire1 <- ISODomainConsistency$new()
cr_inspire1 <- ISOConformanceResult$new()
cr_inspire_spec1 <- ISOCitation$new()
cr_title1 <- paste(
 "Commission Regulation (EU) No 1089/2010 of 23 November 2010 implementing Directive 2007/2/EC",
 "of the European Parliament and of the Council as regards interoperability of spatial data",
   "sets and services"
)
cr_inspire_spec1$setTitle(cr_title1)
cr_inspire1$setExplanation("See the referenced specification")
cr_inspire_date1 <- ISODate$new()
cr_inspire_date1$setDate(ISOdate(2010,12,8))
cr_inspire_date1$setDateType("publication")
cr_inspire_spec1$addDate(cr_inspire_date1)
cr_inspire1$setSpecification(cr_inspire_spec1)
cr_inspire1$setPass(TRUE)
dc_inspire1$addResult(cr_inspire1)
dq$addReport(dc_inspire1)
#INSPIRE - metadata
dc_inspire2 <- ISODomainConsistency$new()
cr_inspire2 <- ISOConformanceResult$new()
cr_inspire_spec2 <- ISOCitation$new()
cr_title2 <- paste(
 "COMMISSION REGULATION (EC) No 1205/2008 of 3 December 2008 implementing Directive 2007/2/EC",
   "of the European Parliament and of the Council as regards metadata"
)
cr_inspire_spec2$setTitle(cr_title2)
cr_inspire2$setExplanation("See the referenced specification")
cr_inspire_date2 <- ISODate$new()
cr_inspire_date2$setDate(ISOdate(2008,12,4))
cr_inspire_date2$setDateType("publication")
cr_inspire_spec2$addDate(cr_inspire_date2)
```

```
cr_inspire2$setSpecification(cr_inspire_spec2)
cr_inspire2$setPass(TRUE)
dc_inspire2$addResult(cr_inspire2)
dq$addReport(dc_inspire2)

#add lineage
lineage <- ISOLineage$new()
lineage$setStatement("statement")
dq$setLineage(lineage)

md$addDataQualityInfo(dq)

#Content Information
#-------------------------
#add a feature catalogue description
fcd <- ISOFeatureCatalogueDescription$new()
fcd$setComplianceCode(FALSE)
fcd$addLanguage("eng")
fcd$setIncludedWithDataset(FALSE)
cit = ISOCitation$new()
cit$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
cit$addDate(d)
cit$setEdition("1.0")
cit$setEditionDate(as.Date(ISOdate(2015, 1, 1, 1)))
contact = ISOContact$new()
fcLink <- ISOOnlineResource$new()
fcLink$setLinkage("http://somelink/featurecatalogue")
contact$setOnlineResource(fcLink)
rp = ISOResponsibleParty$new()
rp$setRole("publisher")
rp$setContactInfo(contact)
cit$addCitedResponsibleParty(rp)
fcd$addFeatureCatalogueCitation(cit)
md$addContentInfo(fcd)

#XML representation of the ISOImageryMetadata
xml <- md$encode()

#example 2 - READ: Create an ISO imagery metadata reading from XML
######################################################

require(XML)
xmlfile <- system.file("extdata/examples", "metadata.xml", package = "geometa")
xml <- xmlParse(xmlfile)
md <- ISOImageryMetadata$new(xml = xml)
```

ISOImageryNominalResolution *ISOImageryNominalResolution*

Description

ISOImageryNominalResolution

ISOImageryNominalResolution

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery nominal resolution

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `ISOImageryNominalResolution`

Public fields

`scanningResolution` scanningResolution [0..1]: ISODistance

`groundResolution` groundResolution [0..1]: ISODistance

Methods

Public methods:

- `ISOImageryNominalResolution$new()`
- `ISOImageryNominalResolution$setScanningResolution()`
- `ISOImageryNominalResolution$setGroundResolution()`
- `ISOImageryNominalResolution$clone()`

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

Usage:

```
ISOImageryNominalResolution$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setScanningResolution()`**:** Set scanning resolution

Usage:

```
ISOImageryNominalResolution$setScanningResolution(resolution)
```

Arguments:

`resolution` object of class ISODistance

Method `setGroundResolution():` Set ground resolution

Usage:

```
ISOImageryNominalResolution$setGroundResolution(resolution)
```

Arguments:

`resolution` object of class ISODistance

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#encoding
dq <- ISOImageryNominalResolution$new()
d <- ISODistance$new(value = 1, uom = "m", useUomURI = TRUE)
dq$setScanningResolution(d)
dq$setGroundResolution(d)

#xml
xml <- dq$encode()
```

`ISOImageryObjective` *ISOImageryObjective*

Description

ISOImageryObjective

ISOImageryObjective

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery objective

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryObjective`

Public fields

`identifier` identifier [1..1]: ISOMetaIdentifier

`priority` priority [0..1]: character|ISOLocalisedCharacterString

`type` type [0..*]: ISOImageryObjectiveType

`function` function [0..*]: character|ISOLocalisedCharacterString

`extent` extent [0..*]: ISOExtent

`sensingInstrument` sensingInstrument [0..*]: ISOImageryInstrument

`pass` pass [0..*]: ISOImageryPlatformPass

`objectiveOccurance` objectiveOccurance [1..*]: ISOImageryEvent

Methods

Public methods:

- `ISOImageryObjective$new()`
- `ISOImageryObjective$setIdentifier()`
- `ISOImageryObjective$setPriority()`
- `ISOImageryObjective$addType()`
- `ISOImageryObjective$delType()`
- `ISOImageryObjective$addFunction()`
- `ISOImageryObjective$delFunction()`
- `ISOImageryObjective$addExtent()`
- `ISOImageryObjective$delExtent()`
- `ISOImageryObjective$addSensingInstrument()`
- `ISOImageryObjective$delSensingInstrument()`
- `ISOImageryObjective$addPlatformPass()`
- `ISOImageryObjective$delPlatformPass()`
- `ISOImageryObjective$addObjectiveOccurance()`
- `ISOImageryObjective$delObjectiveOccurance()`
- `ISOImageryObjective$clone()`

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

Usage:

```
ISOImageryObjective$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setIdentifier()`: Set identifier

Usage:

```
ISOImageryObjective$setIdentifier(identifier)
```

Arguments:

`identifier` object of class ISOMetaIdentifier or character

Method `setPriority()`: Set priority

Usage:

```
ISOImageryObjective$setPriority(priority, locales = NULL)
```

Arguments:

`priority` priority

`locales` list of localized texts. Default is `NULL`

Method `addType()`: Adds type

Usage:

```
ISOImageryObjective$addType(type)
```

Arguments:

`type` object of class ISOImageryObjectiveType or any character among values returned by `ISOImageryObjectiveType$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delType()`: Deletes type

Usage:

```
ISOImageryObjective$delType(type)
```

Arguments:

`type` object of class ISOImageryObjectiveType or any character among values returned by `ISOImageryObjectiveType$values()`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addFunction()`: Adds function

Usage:

```
ISOImageryObjective$addFunction(fun, locales = NULL)
```

Arguments:

`fun` fun

`locales` list of localized texts. Default is `NULL`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delFunction()`: Deletes function

Usage:

```
ISOImageryObjective$delFunction(fun, locales = NULL)
```

Arguments:

`fun` fun

`locales` list of localized texts. Default is `NULL`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addExtent():` Adds extent

Usage:

```
ISOImageryObjective$addExtent(extent)
```

Arguments:

`extent` extent, object of class ISOExtent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delExtent():` Deletes extent

Usage:

```
ISOImageryObjective$delExtent(extent)
```

Arguments:

`extent` extent, object of class ISOExtent

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addSensingInstrument():` Adds sensing instrument

Usage:

```
ISOImageryObjective$addSensingInstrument(instrument)
```

Arguments:

`instrument` object of class ISOImageryInstrument

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSensingInstrument():` Deletes sensing instrument

Usage:

```
ISOImageryObjective$delSensingInstrument(instrument)
```

Arguments:

`instrument` object of class ISOImageryInstrument

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addPlatformPass():` Adds platform pass

Usage:

```
ISOImageryObjective$addPlatformPass(pass)
```

Arguments:

`pass` object of class ISOImageryPlatformPass

Returns: `TRUE` if added, `FALSE` otherwise

Method `delPlatformPass():` Deletes platform pass

Usage:

```
ISOImageryObjective$delPlatformPass(pass)
```

Arguments:

`pass` object of class ISOImageryPlatformPass

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addObjectiveOccurance()`**:** Adds objective occurance

Usage:

```
ISOImageryObjective$addObjectiveOccurance(event)
```

Arguments:

`event` object of class ISOImageryEvent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delObjectiveOccurance()`**:** Deletes objective occurance

Usage:

```
ISOImageryObjective$delObjectiveOccurance(event)
```

Arguments:

`event` object of class ISOImageryEvent

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#encoding
md <- ISOImageryObjective$new()
md$setIdentifier("identifier")
md$setPriority("urgent")
md$addType("survey")
md$addFunction("my_function")
evt <- ISOImageryEvent$new()
evt$setIdentifier("event_1")
evt$setTrigger("manual")
evt$setContext("pass")
evt$setSequence("instantaneous")
```

```
evt$setTime(Sys.time())
md$addObjectiveOccurance(evt)
extent <- ISOExtent$new()
bbox <- ISOGeographicBoundingBox$new(minx = -180, miny = -90, maxx = 180, maxy = 90)
extent$addGeographicElement(bbox)
time <- ISOTemporalExtent$new()
start <- ISOdate(2000, 1, 12, 12, 59, 45)
end <- ISOdate(2010, 8, 22, 13, 12, 43)
tp <- GMLTimePeriod$new(beginPosition = start, endPosition = end)
time$setTimePeriod(tp)
extent$addTemporalElement(time)
vert <- ISOVerticalExtent$new()
vert$setMinimumValue(0)
vert$setMaximumValue(19)
extent$addVerticalElement(vert)
md$addExtent(extent)
md$sensingInstrument = NA
md$pass = NA
xml <- md$encode()
```

`ISOImageryObjectiveType` *ISOImageryObjectiveType*

Description

ISOImageryObjectiveType

ISOImageryObjectiveType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery ObjectiveType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOImageryObjectiveType`

Methods

Public methods:

- `ISOImageryObjectiveType$new()`
- `ISOImageryObjectiveType$clone()`

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

Usage:

```
ISOImageryObjectiveType$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#possible values
values <- ISOImageryObjectiveType$values(labels = TRUE)

#some def
survey <- ISOImageryObjectiveType$new(value = "survey")
```

`ISOImageryOperation` *ISOImageryOperation*

Description

ISOImageryOperation

ISOImageryOperation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery Operation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryOperation`

Public fields

`description` description [0..1]: character|ISOLocalisedCharacterString

`citation` citation [0..1]: ISOCitation

`identifier` identifier [1..1]: ISOMetaIdentifier

`status` status [1..1]: ISOStatus

`type` type [0..1]: ISOImageryOperationType

`parentOperation` parentOperation [1..1]: ISOImageryOperation

`childOperation` childOperation [0..*]: ISOImageryOperation

`platform` platform [0..*]: ISOImageryPlatform

`objective` objective [0..*]: ISOImageryObjective

`plan` plan [0..1]: ISOImageryPlan

`significantEvent` significantEvent [0..*]: ISOImageryEvent

Methods

Public methods:

- `ISOImageryOperation$new()`
- `ISOImageryOperation$setDescription()`
- `ISOImageryOperation$setCitation()`
- `ISOImageryOperation$setIdentifier()`
- `ISOImageryOperation$setStatus()`
- `ISOImageryOperation$setType()`
- `ISOImageryOperation$setParentOperation()`
- `ISOImageryOperation$addChildOperation()`
- `ISOImageryOperation$delChildOperation()`
- `ISOImageryOperation$addPlatform()`
- `ISOImageryOperation$delPlatform()`
- `ISOImageryOperation$addObjective()`
- `ISOImageryOperation$delObjective()`
- `ISOImageryOperation$setPlan()`
- `ISOImageryOperation$addSignificantEvent()`
- `ISOImageryOperation$delSignificantEvent()`
- `ISOImageryOperation$clone()`

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

Usage:

```
ISOImageryOperation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setDescription()`: Set description

Usage:

```
ISOImageryOperation$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` list of localized texts. Default is `NULL`

Method `setCitation()`: Set citation

Usage:

```
ISOImageryOperation$setCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Method `setIdentifier()`: Set identifier

Usage:

```
ISOImageryOperation$setIdentifier(identifier)
```

Arguments:

`identifier` object of class ISOMetaIdentifier or character

Method `setStatus()`: Set status

Usage:

```
ISOImageryOperation$setStatus(status)
```

Arguments:

`status` object of class ISOStatus or any character among values returned by `ISOStatus$values()`

Method `setType()`: Set type

Usage:

```
ISOImageryOperation$setType(type)
```

Arguments:

`type` object of class ISOImageryOperationType or any character among values returned by `ISOImageryOperationType$values()`

Method `setParentOperation()`: Set parent operation

Usage:

```
ISOImageryOperation$setParentOperation(operation)
```

Arguments:

`operation` object of class ISOImageryOperation

Method `addChildOperation()`: Adds child operation

Usage:

```
ISOImageryOperation$addChildOperation(operation)
```

Arguments:

`operation` object of class ISOImageryOperation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delChildOperation():` Deletes child operation

Usage:

```
ISOImageryOperation$delChildOperation(operation)
```

Arguments:

`operation` object of class ISOImageryOperation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addPlatform():` Adds platform

Usage:

```
ISOImageryOperation$addPlatform(platform)
```

Arguments:

`platform` object of class ISOImageryPlatform

Returns: `TRUE` if added, `FALSE` otherwise

Method `delPlatform():` Deletes platform

Usage:

```
ISOImageryOperation$delPlatform(platform)
```

Arguments:

`platform` object of class ISOImageryPlatform

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addObjective():` Adds objective

Usage:

```
ISOImageryOperation$addObjective(objective)
```

Arguments:

`objective` object of class ISOImageryObjective

Returns: `TRUE` if added, `FALSE` otherwise

Method `delObjective():` Deletes objective

Usage:

```
ISOImageryOperation$delObjective(objective)
```

Arguments:

`objective` object of class ISOImageryObjective

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setPlan():` Set plan

Usage:

`ISOImageryOperation$setPlan(plan)`

Arguments:

`plan` object of class ISOImageryPlan

Method `addSignificantEvent()`**:** Adds significative event

Usage:

`ISOImageryOperation$addSignificantEvent(event)`

Arguments:

`event` object of class ISOImageryEvent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSignificantEvent()`**:** Deletes significative event

Usage:

`ISOImageryOperation$delSignificantEvent(event)`

Arguments:

`event` object of class ISOImageryEvent

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOImageryOperation$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

---

`ISOImageryOperationType` *ISOImageryOperationType*

Description

ISOImageryOperationType

ISOImageryOperationType

Format

R6Class object.

Value

Object of R6Class for modelling an ISO Imagery Operation type

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISOCodeListValue -> ISOImageryOperationType

Methods

Public methods:

- ISOImageryOperationType$new()
- ISOImageryOperationType$clone()

Method `new()`: Initializes object

Usage:

```
ISOImageryOperationType$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#possible values
values <- ISOImageryOperationType$values(labels = TRUE)

#some def
real <- ISOImageryOperationType$new(value = "real")
```

`ISOImageryPlan` *ISOImageryPlan*

Description

ISOImageryPlan

ISOImageryPlan

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery Plan

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryPlan`

Public fields

`type` type [0..1]: ISOImageryGeometryType

`status` status [1..1]: ISOProgress

`citation` citation [1..1]: ISOCitation

`operation` operation [0..*]: ISOImageryOperation

`satisfiedRequirement` satisfiedRequirement [0..*]: ISOImageryRequirement

Methods

Public methods:

- `ISOImageryPlan$new()`
- `ISOImageryPlan$setType()`
- `ISOImageryPlan$setStatus()`
- `ISOImageryPlan$setCitation()`
- `ISOImageryPlan$addOperation()`
- `ISOImageryPlan$delOperation()`
- `ISOImageryPlan$addSatisfiedRequirement()`
- `ISOImageryPlan$delSatisfiedRequirement()`
- `ISOImageryPlan$clone()`

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

Usage:

```
ISOImageryPlan$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setType()`: Set type

Usage:

```
ISOImageryPlan$setType(type)
```

Arguments:

`type` object of class ISOImageryGeometryType or any character among values returned by `ISOImageryGeometryType$values()`

Method `setStatus()`: Set status

Usage:

```
ISOImageryPlan$setStatus(status)
```

Arguments:

`status` object of class ISOStatus or any character among values returned by `ISOStatus$values()`

Method `setCitation()`: Set citation

Usage:

```
ISOImageryPlan$setCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Method `addOperation()`: Adds operation

Usage:

```
ISOImageryPlan$addOperation(operation)
```

Arguments:

`operation` object of class ISOImageryOperation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delOperation()`: Deletes operation

Usage:

```
ISOImageryPlan$delOperation(operation)
```

Arguments:

`operation` object of class ISOImageryOperation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addSatisfiedRequirement()`: Adds satisfied requirement

Usage:

```
ISOImageryPlan$addSatisfiedRequirement(requirement)
```

Arguments:

`requirement` object of class ISOImageryRequirement

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSatisfiedRequirement()`**:** Deletes satisfied requirement

Usage:

```
ISOImageryPlan$delSatisfiedRequirement(requirement)
```

Arguments:

`requirement` object of class ISOImageryRequirement

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryPlan$new()
md$setType("point")
md$setStatus("completed")

#add citation
rp1 <- ISOResponsibleParty$new()
rp1$setIndividualName("someone1")
rp1$setOrganisationName("somewhere1")
rp1$setPositionName("someposition1")
rp1$setRole("pointOfContact")
contact1 <- ISOContact$new()
phone1 <- ISOTelephone$new()
phone1$setVoice("myphonenumber1")
phone1$setFacsimile("myfacsimile1")
contact1$setPhone(phone1)
address1 <- ISOAddress$new()
address1$setDeliveryPoint("theaddress1")
address1$setCity("thecity1")
address1$setPostalCode("111")
address1$setCountry("France")
address1$setEmail("someone1@theorg.org")
contact1$setAddress(address1)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
```

```
contact1$setOnlineResource(res)
rp1$setContactInfo(contact1)

#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(ISOdate(2015,1,1))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp1)
md$setCitation(ct)
xml <- md$encode()
```

`ISOImageryPlatform` *ISOImageryPlatform*

Description

ISOImageryPlatform

ISOImageryPlatform

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery platform

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryPlatform`

Public fields

`citation` citation [0..*]: ISOCitation

`identifier` identifier [1..1]: ISOMetaIdentifier

`description` description [0..1]: character|ISOLocalisedCharacterString

`sponsor` sponsor [0..*]: ISOResponsibleParty

`instrument` instrument [0..*]: ISOImageryInstrument

Methods

Public methods:

- `ISOImageryPlatform$new()`
- `ISOImageryPlatform$addCitation()`
- `ISOImageryPlatform$delCitation()`
- `ISOImageryPlatform$setIdentifier()`
- `ISOImageryPlatform$setDescription()`
- `ISOImageryPlatform$addSponsor()`
- `ISOImageryPlatform$delSponsor()`
- `ISOImageryPlatform$addInstrument()`
- `ISOImageryPlatform$delInstrument()`
- `ISOImageryPlatform$clone()`

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

Usage:

```
ISOImageryPlatform$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addCitation()`**:** Adds citation

Usage:

```
ISOImageryPlatform$addCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delCitation()`**:** Deletes citation

Usage:

```
ISOImageryPlatform$delCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setIdentifier()`**:** Set identifier

Usage:

```
ISOImageryPlatform$setIdentifier(identifier)
```

Arguments:

`identifier` object of class ISOMetaIdentifier or character

Method `setDescription()`**:** Set description

Usage:

```
ISOImageryPlatform$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` list of localized texts. Default is `NULL`

Method `addSponsor():` Adds sponsor

Usage:

```
ISOImageryPlatform$addSponsor(sponsor)
```

Arguments:

`sponsor` object of class ISOResponsibleParty

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSponsor():` Deletes sponsor

Usage:

```
ISOImageryPlatform$delSponsor(sponsor)
```

Arguments:

`sponsor` object of class ISOResponsibleParty

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addInstrument():` Adds instrument

Usage:

```
ISOImageryPlatform$addInstrument(instrument)
```

Arguments:

`instrument` object of class ISOImageryInstrument

Returns: `TRUE` if added, `FALSE` otherwise

Method `delInstrument():` Deletes instrument

Usage:

```
ISOImageryPlatform$delInstrument(instrument)
```

Arguments:

`instrument` object of class ISOImageryInstrument

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryPlatform$new()

#add citation
rp1 <- ISOResponsibleParty$new()
rp1$setIndividualName("someone1")
rp1$setOrganisationName("somewhere1")
rp1$setPositionName("someposition1")
rp1$setRole("pointOfContact")
contact1 <- ISOContact$new()
phone1 <- ISOTelephone$new()
phone1$setVoice("myphonenumber1")
phone1$setFacsimile("myfacsimile1")
contact1$setPhone(phone1)
address1 <- ISOAddress$new()
address1$setDeliveryPoint("theaddress1")
address1$setCity("thecity1")
address1$setPostalCode("111")
address1$setCountry("France")
address1$setEmail("someone1@theorg.org")
contact1$setAddress(address1)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact1$setOnlineResource(res)
rp1$setContactInfo(contact1)

#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(ISOdate(2015,1,1))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp1)
md$addCitation(ct)

md$setIdentifier("identifier")
md$setDescription("some description")

xml <- md$encode()
```

ISOImageryPlatformPass *ISOImageryPlatformPass*

Description

ISOImageryPlatformPass

ISOImageryPlatformPass

Format

R6Class object.

Value

Object of R6Class for modelling an ISO imagery PlatformPass

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> ISOImageryPlatformPass

Public fields

identifier identifier [1..1]: ISOMetaIdentifier

extent extent [0..1]: ?

relatedEvent relatedEvent [0..*]: ISOImageryEvent

Methods

Public methods:

- ISOImageryPlatformPass$new()
- ISOImageryPlatformPass$setIdentifier()
- ISOImageryPlatformPass$setExtent()
- ISOImageryPlatformPass$addEvent()
- ISOImageryPlatformPass$delEvent()
- ISOImageryPlatformPass$clone()

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

Usage:

```
ISOImageryPlatformPass$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setIdentifier()`**:** Set identifier

Usage:

```
ISOImageryPlatformPass$setIdentifier(identifier)
```

Arguments:

`identifier` object of class ISOMetaIdentifier or character

Method `setExtent():` Set extent

Usage:

```
ISOImageryPlatformPass$setExtent(extent)
```

Arguments:

`extent` simple feature geometry object from **sf**

Method `addEvent():` Adds event

Usage:

```
ISOImageryPlatformPass$addEvent(event)
```

Arguments:

`event` object of class ISOImageryEvent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delEvent():` Deletes event

Usage:

```
ISOImageryPlatformPass$delEvent(event)
```

Arguments:

`event` object of class ISOImageryEvent

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryPlatformPass$new()
md$setIdentifier("identifier")

require(sf)
outer = matrix(c(0,0,10,0,10,10,0,10,0,0),ncol=2, byrow=TRUE)
hole1 = matrix(c(1,1,1,2,2,2,2,1,1,1),ncol=2, byrow=TRUE)
hole2 = matrix(c(5,5,5,6,6,6,6,5,5,5),ncol=2, byrow=TRUE)
pts = list(outer, hole1, hole2)
pl = st_polygon(pts)
md$setExtent(pl)

xml <- md$encode()
```

ISOImageryPolarisationOrientation *ISOImageryPolarisationOrientation*

Description

ISOImageryPolarisationOrientation

ISOImageryPolarisationOrientation

Format

R6Class object.

Value

Object of R6Class for modelling an ISO Imagery Polarisation orientation

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISOCodeListValue -> ISOImageryPolarisationOrientation

Methods

Public methods:

- ISOImageryPolarisationOrientation$new()
- ISOImageryPolarisationOrientation$clone()

Method new()**:** Initializes object

Usage:

```
ISOImageryPolarisationOrientation$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOImageryPolarisationOrientation$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#possible values
values <- ISOImageryPolarisationOrientation$values(labels = TRUE)

#some def
h <- ISOImageryPolarisationOrientation$new(value = "horizontal")
```

`ISOImageryPriority` *ISOImageryPriority*

Description

ISOImageryPriority

ISOImageryPriority

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery priority

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOImageryPriority`

Methods

Public methods:

- `ISOImageryPriority$new()`
- `ISOImageryPriority$clone()`

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

Usage:

```
ISOImageryPriority$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#possible values
values <- ISOImageryPriority$values(labels = TRUE)

#some def
highImp <- ISOImageryPriority$new(value = "highImportance")
```

`ISOImageryProcessing` *ISOImageryProcessing*

Description

ISOImageryProcessing

ISOImageryProcessing

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery processing

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryProcessing`

Public fields

`identifier` identifier [1..1]: ISOMetaIdentifier

`softwareReference` softwareReference [0.1]: ISOCitation

`procedureDescription` procedureDescription [0..1]: character|ISOLocalisedCharacterString

`documentation` documentation [0..*]: ISOCitation

`runTimeParameters` runTimeParameters [0..1]: character

`algorithm` algorithm [0..*]: ISOImageryAlgorithm

Methods

Public methods:

- `ISOImageryProcessing$new()`
- `ISOImageryProcessing$setIdentifier()`
- `ISOImageryProcessing$addSoftwareReference()`
- `ISOImageryProcessing$delSoftwareReference()`
- `ISOImageryProcessing$setProcedureDescription()`
- `ISOImageryProcessing$addDocumentation()`
- `ISOImageryProcessing$delDocumentation()`
- `ISOImageryProcessing$setRunTimeParameters()`
- `ISOImageryProcessing$addAlgorithm()`
- `ISOImageryProcessing$delAlgorithm()`
- `ISOImageryProcessing$clone()`

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

Usage:

```
ISOImageryProcessing$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setIdentifier()`**:** Set identifier

Usage:

```
ISOImageryProcessing$setIdentifier(identifier)
```

Arguments:

`identifier` object of class ISOMetaIdentifier or character

Method `addSoftwareReference()`**:** Adds software reference

Usage:

```
ISOImageryProcessing$addSoftwareReference(softwareReference)
```

Arguments:

`softwareReference` object of class ISOCitation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSoftwareReference()`**:** Deletes software reference

Usage:

```
ISOImageryProcessing$delSoftwareReference(softwareReference)
```

Arguments:

`softwareReference` object of class ISOCitation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setProcedureDescription()`**:** Set procedure description

Usage:

```
ISOImageryProcessing$setProcedureDescription(
  procedureDescription,
  locales = NULL
)
```

Arguments:

`procedureDescription` procedure description

`locales` list of localized texts. Default is `NULL`

Method `addDocumentation()`**:** Adds documentation

Usage:

```
ISOImageryProcessing$addDocumentation(documentation)
```

Arguments:

`documentation` object of class ISOCitation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDocumentation()`**:** Deletes documentation

Usage:

```
ISOImageryProcessing$delDocumentation(documentation)
```

Arguments:

`documentation` object of class ISOCitation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setRunTimeParameters()`**:** Set runtime parameters

Usage:

```
ISOImageryProcessing$setRunTimeParameters(params)
```

Arguments:

`params` parameters

Method `addAlgorithm()`**:** Adds algorithm

Usage:

```
ISOImageryProcessing$addAlgorithm(algorithm)
```

Arguments:

`algorithm` object of class ISOImageryAlgorithm

Returns: `TRUE` if added, `FALSE` otherwise

Method `delAlgorithm()`**:** Deletes algorithm

Usage:

```
ISOImageryProcessing$delAlgorithm(algorithm)
```

Arguments:

`algorithm` object of class ISOImageryAlgorithm

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryProcessing$new()

#add citation
rp1 <- ISOResponsibleParty$new()
rp1$setIndividualName("someone1")
rp1$setOrganisationName("somewhere1")
rp1$setPositionName("someposition1")
rp1$setRole("pointOfContact")
contact1 <- ISOContact$new()
phone1 <- ISOTelephone$new()
phone1$setVoice("myphonenumber1")
phone1$setFacsimile("myfacsimile1")
contact1$setPhone(phone1)
address1 <- ISOAddress$new()
address1$setDeliveryPoint("theaddress1")
address1$setCity("thecity1")
address1$setPostalCode("111")
address1$setCountry("France")
address1$setEmail("someone1@theorg.org")
contact1$setAddress(address1)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact1$setOnlineResource(res)
rp1$setContactInfo(contact1)

#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(ISOdate(2015,1,1))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp1)

md$setIdentifier("identifier")
md$setProcedureDescription("some description")
md$addSoftwareReference(ct)
md$addDocumentation(ct)
md$setRunTimeParameters("params")

xml <- md$encode()
```

ISOImageryProcessStep *ISOImageryProcessStep*

Description

ISOImageryProcessStep

ISOImageryProcessStep

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery process step

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOProcessStep` -> `ISOImageryProcessStep`

Public fields

`processingInformation` processingInformation [0..1]: ISOImageryProcessing

`output` output [0..*]: list of ISOImagerySource

`report` report [0..*]: list of ISOImageryProcessStepReport

Methods

Public methods:

- `ISOImageryProcessStep$new()`
- `ISOImageryProcessStep$setProcessingInformation()`
- `ISOImageryProcessStep$addOutput()`
- `ISOImageryProcessStep$delOutput()`
- `ISOImageryProcessStep$addReport()`
- `ISOImageryProcessStep$delReport()`
- `ISOImageryProcessStep$clone()`

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

Usage:

```
ISOImageryProcessStep$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setProcessingInformation()`**:** Set processing info

Usage:

```
ISOImageryProcessStep$setProcessingInformation(processingInfo)
```

Arguments:

`processingInfo` object of class ISOImageryProcessing

Method `addOutput()`**:** Adds output

Usage:

`ISOImageryProcessStep$addOutput(output)`

Arguments:

`output` object of class ISOImagerySource

Returns: `TRUE` if added, `FALSE` otherwise

Method `delOutput()`**:** Deletes output

Usage:

`ISOImageryProcessStep$delOutput(output)`

Arguments:

`output` object of class ISOImagerySource

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addReport()`**:** Adds report

Usage:

`ISOImageryProcessStep$addReport(report)`

Arguments:

`report` object of class ISOImageryProcessStepReport

Returns: `TRUE` if added, `FALSE` otherwise

Method `delReport()`**:** Deletes report

Usage:

`ISOImageryProcessStep$delReport(report)`

Arguments:

`report` object of class ISOImageryProcessStepReport

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOImageryProcessStep$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
ps <- ISOImageryProcessStep$new()
ps$setDescription("description")
ps$setRationale("rationale")
ps$setDateTime( ISOdate(2015, 1, 1, 23, 59, 59))
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone") #and more responsible party properties..
ps$addProcessor(rp)

#specific methods to ISO 19115-2
process <- ISOImageryProcessing$new()

#add citation
rp1 <- ISOResponsibleParty$new()
rp1$setIndividualName("someone1")
rp1$setOrganisationName("somewhere1")
rp1$setPositionName("someposition1")
rp1$setRole("pointOfContact")
contact1 <- ISOContact$new()
phone1 <- ISOTelephone$new()
phone1$setVoice("myphonenumber1")
phone1$setFacsimile("myfacsimile1")
contact1$setPhone(phone1)
address1 <- ISOAddress$new()
address1$setDeliveryPoint("theaddress1")
address1$setCity("thecity1")
address1$setPostalCode("111")
address1$setCountry("France")
address1$setEmail("someone1@theorg.org")
contact1$setAddress(address1)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact1$setOnlineResource(res)
rp1$setContactInfo(contact1)

#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(ISOdate(2015,1,1))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp1)

process$setIdentifier("identifier")
process$setProcedureDescription("some description")
process$addSoftwareReference(ct)
```

```
process$addDocumentation(ct)
process$setRunTimeParameters("params")
ps$setProcessingInformation(process)

#output
trg <- ISOImagerySource$new()
trg$setProcessedLevel("level")
res <- ISOImageryNominalResolution$new()
d <- ISODistance$new(value = 1, uom = "m", useUomURI = TRUE)
res$setScanningResolution(d)
trg$setResolution(res)
ps$addOutput(trg)

#report
rep <- ISOImageryProcessStepReport$new()
rep$setName("report")
rep$setDescription("description")
rep$setFileType("filetype")
ps$addReport(rep)

xml <- ps$encode()
```

`ISOImageryProcessStepReport` *ISOImageryProcessStepReport*

Description

ISOImageryProcessStepReport

ISOImageryProcessStepReport

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery ProcessStepReport

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryProcessStepReport`

Public fields

`name` name [1..1]: character|ISOLocalisedCharacterString

`description` description [0..1]: character|ISOLocalisedCharacterString

`fileType` fileType [0..1]: character|ISOLocalisedCharacterString

Methods

Public methods:

- `ISOImageryProcessStepReport$new()`
- `ISOImageryProcessStepReport$setName()`
- `ISOImageryProcessStepReport$setDescription()`
- `ISOImageryProcessStepReport$setFileType()`
- `ISOImageryProcessStepReport$clone()`

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

Usage:

```
ISOImageryProcessStepReport$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setName()`**:** Set name

Usage:

```
ISOImageryProcessStepReport$setName(name, locales = NULL)
```

Arguments:

`name` name

`locales` list of localized texts. Default is `NULL`

Method `setDescription()`**:** Set description

Usage:

```
ISOImageryProcessStepReport$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` list of localized texts. Default is `NULL`

Method `setFileType()`**:** Set file type

Usage:

```
ISOImageryProcessStepReport$setFileType(fileType, locales = NULL)
```

Arguments:

`fileType` file type

`locales` list of localized texts. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryProcessStepReport$new()
md$setName("my_report")
md$setDescription("description")
md$setFileType("md")
xml <- md$encode()
```

ISOImageryRangeElementDescription
ISOImageryRangeElementDescription

Description

ISOImageryRangeElementDescription

ISOImageryRangeElementDescription

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOImageryRangeElementDescription

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryRangeElementDescription`

Public fields

`name` name [0..1] : character

`definition` definition [0..1] : character

`rangeElement` rangeElement [0..*] : ISORecord

Methods

Public methods:

- `ISOImageryRangeElementDescription$new()`
- `ISOImageryRangeElementDescription$setName()`
- `ISOImageryRangeElementDescription$setDefinition()`
- `ISOImageryRangeElementDescription$addRangeElement()`
- `ISOImageryRangeElementDescription$delRangeElement()`
- `ISOImageryRangeElementDescription$clone()`

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

Usage:

```
ISOImageryRangeElementDescription$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setName()`**:** Set name

Usage:

```
ISOImageryRangeElementDescription$setName(name, locales = NULL)
```

Arguments:

`name` name

`locales` list of localized texts. Default is `NULL`

Method `setDefinition()`**:** Set definition

Usage:

```
ISOImageryRangeElementDescription$setDefinition(definition, locales = NULL)
```

Arguments:

`definition` definition

`locales` list of localized texts. Default is `NULL`

Method `addRangeElement()`**:** Adds range element

Usage:

```
ISOImageryRangeElementDescription$addRangeElement(record)
```

Arguments:

`record` object of class ISORecord or character

Returns: `TRUE` if added, `FALSE` otherwise

Method `delRangeElement()`**:** Deletes range element

Usage:

```
ISOImageryRangeElementDescription$delRangeElement(record)
```

Arguments:

`record` object of class ISORecord or character

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOImageryRangeElementDescription$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#create object
md <- ISOImageryRangeElementDescription$new()
md$setName("name")
md$setDefinition("description")
md$addRangeElement("record1")
md$addRangeElement("record2")
xml <- md$encode()
```

`ISOImageryRequestedDate` *ISOImageryRequestedDate*

Description

ISOImageryRequestedDate

ISOImageryRequestedDate

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery requested date

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryRequestedDate`

Public fields

`requestedDateOfCollection` requestedDateOfCollection

`latestAcceptableDate` latestAcceptableDate

Methods

Public methods:

- `ISOImageryRequestedDate$new()`
- `ISOImageryRequestedDate$setRequestedDateOfCollection()`
- `ISOImageryRequestedDate$setLatestAcceptableDate()`
- `ISOImageryRequestedDate$clone()`

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

Usage:

```
ISOImageryRequestedDate$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setRequestedDateOfCollection()`**:** Set requested date of collection

Usage:

```
ISOImageryRequestedDate$setRequestedDateOfCollection(date)
```

Arguments:

`date` object of class POSIXct

Method `setLatestAcceptableDate()`**:** Set latest acceptable date

Usage:

```
ISOImageryRequestedDate$setLatestAcceptableDate(date)
```

Arguments:

`date` object of class POSIXct

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#create band range dimension
md <- ISOImageryRequestedDate$new()
md$setRequestedDateOfCollection(Sys.time())
md$setLatestAcceptableDate(Sys.time())
xml <- md$encode()
```

ISOImageryRequirement *ISOImageryRequirement*

Description

ISOImageryRequirement

ISOImageryRequirement

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery requirement

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImageryRequirement`

Public fields

`citation` citation [1..1]: ISOCitation

`identifier` identifier [1..1]: ISOMetaIdentifier

`requestor` requestor [0..*]: ISOResponsibleParty

`recipient` recipient [0..*]: ISOResponsibleParty

`priority` priority [1..1]: ISOImageryPriority

`requestedDate` requestedDate [1..1]: ISOImageryRequestedDate

`expiryDate` expiryDate [1..1]: POSIXt

`satisfiedPlan` satisfiedPlan [0..*]: ISOImageryPlan

Methods

Public methods:

- `ISOImageryRequirement$new()`
- `ISOImageryRequirement$setCitation()`
- `ISOImageryRequirement$setIdentifier()`
- `ISOImageryRequirement$addRequestor()`
- `ISOImageryRequirement$delRequestor()`
- `ISOImageryRequirement$addRecipient()`
- `ISOImageryRequirement$delRecipient()`
- `ISOImageryRequirement$setPriority()`
- `ISOImageryRequirement$setRequestedDate()`
- `ISOImageryRequirement$setExpiryDate()`
- `ISOImageryRequirement$addSatisfiedPlan()`
- `ISOImageryRequirement$delSatisfiedPlan()`
- `ISOImageryRequirement$clone()`

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

Usage:

```
ISOImageryRequirement$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setCitation()`**:** Set citation

Usage:

```
ISOImageryRequirement$setCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Method `setIdentifier()`**:** Set identifier

Usage:

```
ISOImageryRequirement$setIdentifier(identifier)
```

Arguments:

`identifier` object of class ISOMetaIdentifier or character

Method `addRequestor()`**:** Adds requestor

Usage:

```
ISOImageryRequirement$addRequestor(requestor)
```

Arguments:

`requestor` object of class ISOResponsibleParty

Returns: `TRUE` if added, `FALSE` otherwise

Method `delRequestor()`**:** Deletes requestor

Usage:

```
ISOImageryRequirement$delRequestor(requestor)
```

Arguments:

`requestor` object of class ISOResponsibleParty

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addRecipient()`: Adds recipient

Usage:

```
ISOImageryRequirement$addRecipient(recipient)
```

Arguments:

`recipient` object of class ISOResponsibleParty

Returns: `TRUE` if added, `FALSE` otherwise

Method `delRecipient()`: Deletes recipient

Usage:

```
ISOImageryRequirement$delRecipient(recipient)
```

Arguments:

`recipient` object of class ISOResponsibleParty

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setPriority()`: Set priority

Usage:

```
ISOImageryRequirement$setPriority(priority)
```

Arguments:

`priority` object of class ISOImageryPriority pr any character among values returned by `ISOImageryPriority$values(`

Method `setRequestedDate()`: Set requested date

Usage:

```
ISOImageryRequirement$setRequestedDate(date)
```

Arguments:

`date` object of class ISOImageryRequestedDate

Method `setExpiryDate()`: Set expiry date

Usage:

```
ISOImageryRequirement$setExpiryDate(date)
```

Arguments:

`date` object of class POSIXct

Method `addSatisfiedPlan()`: Adds satisfied plan

Usage:

```
ISOImageryRequirement$addSatisfiedPlan(plan)
```

Arguments:

`plan` object of class ISOImageryPlan

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSatisfiedPlan()`**:** Deletes satisfied plan

Usage:

```
ISOImageryRequirement$delSatisfiedPlan(plan)
```

Arguments:

`plan` object of class ISOImageryPlan

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImageryRequirement$new()
md$setIdentifier("identifier")
#add citation
rp1 <- ISOResponsibleParty$new()
rp1$setIndividualName("someone1")
rp1$setOrganisationName("somewhere1")
rp1$setPositionName("someposition1")
rp1$setRole("pointOfContact")
contact1 <- ISOContact$new()
phone1 <- ISOTelephone$new()
phone1$setVoice("myphonenumber1")
phone1$setFacsimile("myfacsimile1")
contact1$setPhone(phone1)
address1 <- ISOAddress$new()
address1$setDeliveryPoint("theaddress1")
address1$setCity("thecity1")
address1$setPostalCode("111")
address1$setCountry("France")
address1$setEmail("someone1@theorg.org")
contact1$setAddress(address1)
res <- ISOOnlineResource$new()
```

```
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact1$setOnlineResource(res)
rp2 <- ISOResponsibleParty$new()
rp2$setIndividualName("someone2")
rp2$setOrganisationName("somewhere2")
rp2$setPositionName("someposition2")
rp2$setRole("pointOfContact")
contact2 <- ISOContact$new()
phone2 <- ISOTelephone$new()
phone2$setVoice("myphonenumber2")
phone2$setFacsimile("myfacsimile2")
contact1$setPhone(phone2)
address2 <- ISOAddress$new()
address2$setDeliveryPoint("theaddress2")
address2$setCity("thecity2")
address2$setPostalCode("111")
address2$setCountry("France")
address2$setEmail("someone2@theorg.org")
contact2$setAddress(address2)
contact2$setOnlineResource(res)
rp2$setContactInfo(contact2)

#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(ISOdate(2015,1,1))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp1)
md$setCitation(ct)
md$addRequestor(rp1)
md$addRecipient(rp2)
md$setPriority("highImportance")

rd <- ISOImageryRequestedDate$new()
rd$setRequestedDateOfCollection(Sys.time())
rd$setLatestAcceptableDate(Sys.time())
md$setRequestedDate(rd)
md$setExpiryDate(Sys.time())
xml <- md$encode()
```

ISOImagerySensorType *ISOImagerySensorType*

Description

ISOImagerySensorType

ISOImagerySensorType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery sensor type

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImagerySensorType`

Methods

Public methods:

- `ISOImagerySensorType$new()`
- `ISOImagerySensorType$clone()`

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

Usage:

```
ISOImagerySensorType$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImagerySensorType$new()
```

`ISOImagerySequence` *ISOImagerySequence*

Description

ISOImagerySequence

ISOImagerySequence

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery sequence

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOImagerySequence`

Methods

Public methods:

- `ISOImagerySequence$new()`
- `ISOImagerySequence$clone()`

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

Usage:

```
ISOImagerySequence$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#possible values
values <- ISOImagerySequence$values(labels = TRUE)

#some def
inst <- ISOImagerySequence$new(value = "instantaneous")
```

`ISOImagerySource` *ISOImagerySource*

Description

ISOImagerySource

ISOImagerySource

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery source

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOImagerySource`

Public fields

`processedLevel` processedLevel [0..1]: ISOMetaIdentifier

`resolution` resolution [0..1]: ISOImageryNominalResolution

Methods

Public methods:

- `ISOImagerySource$new()`
- `ISOImagerySource$setProcessedLevel()`
- `ISOImagerySource$setResolution()`
- `ISOImagerySource$clone()`

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

Usage:

`ISOImagerySource$new(xml = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

Method `setProcessedLevel():` Set processed level

Usage:

`ISOImagerySource$setProcessedLevel(processedLevel)`

Arguments:

`processedLevel` object of class ISOMetaIdentifier or character

Method `setResolution():` Set resolution

Usage:

`ISOImagerySource$setResolution(resolution)`

Arguments:

`resolution` object of class ISOImageryNominalResolution

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

Usage:

`ISOImagerySource$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
md <- ISOImagerySource$new()
md$setProcessedLevel("identifier")
res <- ISOImageryNominalResolution$new()
d <- ISODistance$new(value = 1, uom = "m", useUomURI = TRUE)
res$setScanningResolution(d)
md$setResolution(res)

xml <- md$encode()
```

`ISOImageryTransferFunctionType` *ISOImageryTransferFunctionType*

Description

ISOImageryTransferFunctionType

ISOImageryTransferFunctionType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery transfer function type

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOImageryTransferFunctionType`

Methods

Public methods:

- `ISOImageryTransferFunctionType$new()`
- `ISOImageryTransferFunctionType$clone()`

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

Usage:

```
ISOImageryTransferFunctionType$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#possible values
values <- ISOImageryTransferFunctionType$values(labels = TRUE)

#some def
log <- ISOImageryTransferFunctionType$new(value = "logarithmic")
```

`ISOImageryTrigger` *ISOImageryTrigger*

Description

ISOImageryTrigger

ISOImageryTrigger

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery trigger

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOImageryTrigger`

Methods

Public methods:

- `ISOImageryTrigger$new()`
- `ISOImageryTrigger$clone()`

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

Usage:

```
ISOImageryTrigger$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

Examples

```
#possible values
values <- ISOImageryTrigger$values(labels = TRUE)

#some def
auto <- ISOImageryTrigger$new(value = "automatic")
```

`ISOImageryUsability` *ISOImageryUsability*

Description

ISOImageryUsability

ISOImageryUsability

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO imagery usability

Methods inherited from `ISODataQualityAbstractElement`

See methods description at `ISODataQualityAbstractElement`

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `ISOImageryUsability`

Methods

Public methods:

- `ISOImageryUsability$new()`
- `ISOImageryUsability$clone()`

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

Usage:

```
ISOImageryUsability$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115-2:2009 - Geographic information – Metadata Part 2: Extensions for imagery and gridded data

---

`ISOImagingCondition` *ISOImagingCondition*

Description

ISOImagingCondition

ISOImagingCondition

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOImagingCondition

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOImagingCondition`

Methods

Public methods:

- `ISOImagingCondition$new()`
- `ISOImagingCondition$clone()`

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

Usage:

```
ISOImagingCondition$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOImagingCondition$values(labels = TRUE)

#ImagingCondition
ImagingCondition <- ISOImagingCondition$new(value = "rain")
```

`ISOInheritanceRelation` *ISOInheritanceRelation*

Description

ISOInheritanceRelation

ISOInheritanceRelation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOInheritanceRelation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOInheritanceRelation`

Public fields

`name` name [0..1]: character

`description` description [0..1]: character

`uniqueInstance` uniqueInstance: logical

`subtype` subtype [1..1]: ISOFeatureType

`supertype` supertype [1..1]: ISOFeatureType

Methods

Public methods:

- `ISOInheritanceRelation$setName()`
- `ISOInheritanceRelation$setDescription()`
- `ISOInheritanceRelation$setUniqueInstance()`
- `ISOInheritanceRelation$setSubtype()`
- `ISOInheritanceRelation$setSupertype()`
- `ISOInheritanceRelation$clone()`

Method `setName()`**:** Set name

Usage:

```
ISOInheritanceRelation$setName(name, locales = NULL)
```

Arguments:

`name` name

`locales` list of localized texts. Default is `NULL`

Method `setDescription()`**:** Set description

Usage:

```
ISOInheritanceRelation$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` list of localized texts. Default is `NULL`

Method `setUniqueInstance()`**:** Set unique instance

Usage:

```
ISOInheritanceRelation$setUniqueInstance(uniqueInstance)
```

Arguments:

`uniqueInstance` object of class logical

Method `setSubtype()`**:** Set sub feature type

Usage:

```
ISOInheritanceRelation$setSubtype(featureType)
```

Arguments:

`featureType` object of class ISOFeatureType

Method `setSupertype()`**:** Set super feature type

Usage:

```
ISOInheritanceRelation$setSupertype(featureType)
```

Arguments:

`featureType` object of class ISOFeatureType

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

ISOInitiative *ISOInitiative*

Description

ISOInitiative

ISOInitiative

Format

R6Class object.

Value

Object of R6Class for modelling an ISOInitiative

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISOAbstractAggregate -> ISOInitiative

Methods

Public methods:

- ISOInitiative$new()
- ISOInitiative$clone()

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

Usage:

```
ISOInitiative$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOInitiativeType` *ISOInitiativeType*

Description

ISOInitiativeType

ISOInitiativeType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO InitiativeType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOInitiativeType`

Methods

Public methods:

- `ISOInitiativeType$new()`
- `ISOInitiativeType$clone()`

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

Usage:

```
ISOInitiativeType$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOInitiativeType$values(labels = TRUE)

#geomOnly
geomOnly <- ISOInitiativeType$new(value = "campaign")
```

`ISOKeywords` *ISOKeywords*

Description

ISOKeywords

ISOKeywords

Format

`R6Class` object.

Value

Object of `R6Class` for modelling a ISO set of keywords

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOKeywords`

Public fields

`keyword` keyword

`type` type

`thesaurusName` thesaurus name

Methods

Public methods:

- `ISOKeywords$new()`
- `ISOKeywords$addKeyword()`
- `ISOKeywords$delKeyword()`
- `ISOKeywords$setKeywordType()`
- `ISOKeywords$setThesaurusName()`

- `ISOKeywords$clone()`

Method `new()`: Initializes object

Usage:

```
ISOKeywords$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addKeyword()`: Adds keyword

Usage:

```
ISOKeywords$addKeyword(keyword, locales = NULL)
```

Arguments:

`keyword` keyword

`locales` list of localized texts. Default is `NULL`

Returns: `TRUE` if added, `FALSe` otherwise

Method `delKeyword()`: Deletes keyword

Usage:

```
ISOKeywords$delKeyword(keyword, locales = NULL)
```

Arguments:

`keyword` keyword

`locales` list of localized texts. Default is `NULL`

Returns: `TRUE` if deleted, `FALSe` otherwise

Method `setKeywordType()`: Set keyword type

Usage:

```
ISOKeywords$setKeywordType(keywordType)
```

Arguments:

`keywordType` object of class ISOKeywordType or any character among values returned by `ISOKeywordType$values()`

Method `setThesaurusName()`: Set thesaurus name

Usage:

```
ISOKeywords$setThesaurusName(thesaurusName)
```

Arguments:

`thesaurusName` object of class ISOCitation

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#a basic keyword set
md <- ISOKeywords$new()
md$addKeyword("keyword1")
md$addKeyword("keyword2")
md$setKeywordType("theme")
th <- ISOCitation$new()
th$setTitle("General")
md$setThesaurusName(th)
xml <- md$encode()

#a keyword set with anchors
md <- ISOKeywords$new()
kwd1 <- ISOAnchor$new(
  name = "keyword1",
  href = "http://myvocabulary.geometa/keyword1"
)
md$addKeyword(kwd1)
kwd2 <- ISOAnchor$new(
  name = "keyword2",
  href = "http://myvocabulary.geometa/keyword2"
)
md$addKeyword(kwd2)
md$setKeywordType("theme")
xml <- md$encode()

#Example for INSPIRE (GEMET Spatial Data Theme)
inspire_kwd <- ISOKeywords$new()
anc1 <- ISOAnchor$new(
  name = "Environmental monitoring facilities",
  href = "http://inspire.ec.europa.eu/theme/ef"
)
inspire_kwd$addKeyword(anc1)
inspire_kwd$setKeywordType("theme")
th <- ISOCitation$new()
th$setTitle(
  ISOAnchor$new(
   name = "GEMET - INSPIRE themes, version 1.0",
   href="http://www.eionet.europa.eu/gemet/inspire_themes"
  )
)
inspire_date <- ISODate$new()
inspire_date$setDate(as.Date("2008-06-01"))
inspire_date$setDateType("publication")
```

```
th$addDate(inspire_date)
inspire_kwd$setThesaurusName(th)
```

`ISOKeywordType` *ISOKeywordType*

Description

ISOKeywordType

ISOKeywordType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO KeywordType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOKeywordType`

Methods

Public methods:

- `ISOKeywordType$new()`
- `ISOKeywordType$clone()`

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

Usage:

```
ISOKeywordType$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOKeywordType$values(labels = TRUE)

#place keywordType
place <- ISOKeywordType$new(value = "place")
```

`ISOLanguage` *ISOLanguage*

Description

ISOLanguage

ISOLanguage

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Language

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOLanguage`

Methods

Public methods:

- `ISOLanguage$new()`
- `ISOLanguage$clone()`

Method `new()`: Initializes object

Usage:

`ISOLanguage$new(xml = NULL, value, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOLanguage$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOLanguage$values(labels = TRUE)

#english language
eng <- ISOLanguage$new(value = "eng")
```

`ISOLegalConstraints` *ISOLegalConstraints*

Description

ISOLegalConstraints

ISOLegalConstraints

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO LegalConstraints

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOConstraints` -> `ISOLegalConstraints`

Public fields

`accessConstraints` accessConstraints [0..*]: ISORestriction

`useConstraints` useConstraints [0..*]: ISORestriction

`otherConstraints` otherConstraints [0..*]: character

Methods

Public methods:

- `ISOLegalConstraints$new()`
- `ISOLegalConstraints$addAccessConstraint()`
- `ISOLegalConstraints$delAccessConstraint()`
- `ISOLegalConstraints$addUseConstraint()`
- `ISOLegalConstraints$delUseConstraint()`
- `ISOLegalConstraints$addOtherConstraint()`
- `ISOLegalConstraints$delOtherConstraint()`
- `ISOLegalConstraints$clone()`

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

Usage:

```
ISOLegalConstraints$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addAccessConstraint()`**:** Adds access constraint

Usage:

```
ISOLegalConstraints$addAccessConstraint(constraint)
```

Arguments:

`constraint` object of class ISORestriction

Returns: `TRUE` if added, `FALSE` otherwise

Method `delAccessConstraint()`**:** Deletes access constraint

Usage:

```
ISOLegalConstraints$delAccessConstraint(constraint)
```

Arguments:

`constraint` object of class ISORestriction

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addUseConstraint()`**:** Adds use constraint

Usage:

```
ISOLegalConstraints$addUseConstraint(constraint)
```

Arguments:

`constraint` object of class ISORestriction

Returns: `TRUE` if added, `FALSE` otherwise

Method `delUseConstraint()`**:** Deletes use constraint

Usage:

`ISOLegalConstraints$delUseConstraint(constraint)`

Arguments:

`constraint` object of class ISORestriction

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addOtherConstraint()`**:** Adds other constraint

Usage:

`ISOLegalConstraints$addOtherConstraint(constraint)`

Arguments:

`constraint` object of class character

Returns: `TRUE` if added, `FALSE` otherwise

Method `delOtherConstraint()`**:** Deletes other constraint

Usage:

`ISOLegalConstraints$delOtherConstraint(constraint)`

Arguments:

`constraint` object of class character

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOLegalConstraints$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create object
md <- ISOLegalConstraints$new()
md$addUseLimitation("limitation1")
md$addUseLimitation("limitation2")
md$addUseLimitation("limitation3")
md$addAccessConstraint("copyright")
```

```
md$addAccessConstraint("license")
md$addUseConstraint("copyright")
md$addUseConstraint("license")

xml <- md$encode()
```

`ISOLength` *ISOLength*

Description

ISOLength

ISOLength

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Length measure

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOMeasure` -> `ISOLength`

Methods

Public methods:

- `ISOLength$new()`
- `ISOLength$clone()`

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

Usage:

`ISOLength$new(xml = NULL, value, uom, useUomURI = FALSE)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`uom` uom symbol of unit of measure used

`useUomURI` use uom URI. Default is `FALSE`

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

Usage:

`ISOLength$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOLineage` *ISOLineage*

Description

ISOLineage

ISOLineage

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Lineage

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOLineage`

Public fields

`statement` statement [0..1]: character

`processStep` processStep [0..*]: ISOProcessStep

`source` source [0..*]: ISOSource

Methods

Public methods:

- `ISOLineage$new()`
- `ISOLineage$setStatement()`
- `ISOLineage$addProcessStep()`
- `ISOLineage$delProcessStep()`
- `ISOLineage$addSource()`
- `ISOLineage$delSource()`
- `ISOLineage$clone()`

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

Usage:

`ISOLineage$new(xml = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

Method `setStatement():` Set statement

Usage:

`ISOLineage$setStatement(statement, locales = NULL)`

Arguments:

`statement` statement

`locales` list of localized texts. Default is `NULL`

Method `addProcessStep():` Adds process step

Usage:

`ISOLineage$addProcessStep(processStep)`

Arguments:

`processStep` object of class ISOProcessStep

Returns: `TRUE` if added, `FALSE` otherwise

Method `delProcessStep():` Deletes process step

Usage:

`ISOLineage$delProcessStep(processStep)`

Arguments:

`processStep` object of class ISOProcessStep

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addSource():` Adds source

Usage:

`ISOLineage$addSource(source)`

Arguments:

`source` object of class ISOSource

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSource():` Deletes source

Usage:

`ISOLineage$delSource(source)`

Arguments:

`source` object of class ISOSource

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOLineage$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
lineage <- ISOLineage$new()
lineage$setStatement("statement")

#add a process step
ps <- ISOProcessStep$new()
ps$setDescription("description")
ps$setRationale("rationale")
ps$setDateTime( ISOdate(2015, 1, 1, 23, 59, 59))
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone") #and more responsible party properties..
ps$addProcessor(rp)
lineage$addProcessStep(ps)

#add a source
src <- ISOSource$new()
src$setDescription("description")
src$setScaleDenominator(1L)
rs <- ISOReferenceSystem$new()
rsId <- ISOReferenceIdentifier$new(code = "4326", codeSpace = "EPSG")
rs$setReferenceSystemIdentifier(rsId)
src$setReferenceSystem(rs)
cit <- ISOCitation$new()
cit$setTitle("sometitle") #and more citation properties...
src$setCitation(cit)
extent <- ISOExtent$new()
bbox <- ISOGeographicBoundingBox$new(minx = -180, miny = -90, maxx = 180, maxy = 90)
extent$addGeographicElement(bbox)
src$addExtent(extent)
lineage$addSource(src)

xml <- lineage$encode()
```

`ISOListedValue` *ISOListedValue*

Description

ISOListedValue

ISOListedValue

Format

R6Class object.

Value

Object of R6Class for modelling an ISOListedValue

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOListedValue`

Public fields

`label` label: character

`code` code [0..1]: character

`definition` definition [0..1]: character

`definitionReference` definitionReference [0..1]: ISODefinitionReference

Methods

Public methods:

- `ISOListedValue$new()`
- `ISOListedValue$setLabel()`
- `ISOListedValue$setCode()`
- `ISOListedValue$setDefinition()`
- `ISOListedValue$setDefinitionReference()`
- `ISOListedValue$clone()`

Method `new()`: Initializes object

Usage:

```
ISOListedValue$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setLabel()`: Set label

Usage:

```
ISOListedValue$setLabel(label, locales = NULL)
```

Arguments:

`label` label

`locales` list of localized texts. Default is `NULL`

Method `setCode()`: Set code

Usage:

```
ISOListedValue$setCode(code, locales = NULL)
```

Arguments:

`code` code

`locales` list of localized texts. Default is `NULL`

Method `setDefinition()`**:** Set definition

Usage:

```
ISOListedValue$setDefinition(definition, locales = NULL)
```

Arguments:

`definition` definition

`locales` list of localized texts. Default is `NULL`

Method `setDefinitionReference()`**:** Set definition reference

Usage:

```
ISOListedValue$setDefinitionReference(definitionReference)
```

Arguments:

`definitionReference` object of class ISODefinitionReference

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

Examples

```
val <- ISOListedValue$new()
val$setCode("code1")
val$setLabel("label1")
val$setDefinition("definition1")
xml <- val$encode()
```

ISOLocale *ISOLocale*

Description

ISOLocale

ISOLocale

Format

R6Class object.

Value

Object of R6Class for modelling an ISO Locale

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> ISOLocale

Public fields

`languageCode` languageCode [1..1]: ISOLanguage

`country` country [0..1]: ISOCountry

`characterEncoding` characterEncoding [1..1]: ISOCharacterSet

Methods

Public methods:

- ISOLocale$new()
- ISOLocale$setId()
- ISOLocale$setLanguage()
- ISOLocale$setCountry()
- ISOLocale$setCharacterSet()
- ISOLocale$clone()

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

Usage:

```
ISOLocale$new(
  xml = NULL,
  id = NULL,
  language = NULL,
  country = NULL,
  encoding = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class

`id` id

`language` language

`country` country

`encoding` encoding

Method `setId()`: Set ID

Usage:

```
ISOLocale$setId(id)
```

Arguments:

`id` id

Method `setLanguage()`: Set language

Usage:

```
ISOLocale$setLanguage(language)
```

Arguments:

`language` object of class ISOLanguage or any character among values returned by `ISOLanguage$values()`

Method `setCountry()`: Set country

Usage:

```
ISOLocale$setCountry(country)
```

Arguments:

`country` object of class ISOCountry or any character among values returned by `ISOCountry$values()` or any other ISO-2 country code

Method `setCharacterSet()`: Set character set

Usage:

```
ISOLocale$setCharacterSet(charset)
```

Arguments:

`charset` object of class ISOCharacterSet or any character among values returned by `ISOCharacterSet$values()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
loc <- ISOLocale$new()
loc$setId("eng")
loc$setLanguage("eng")
loc$setCountry("UK")
loc$setCharacterSet("utf8")
```

`ISOLocaleContainer` *ISOLocaleContainer*

Description

ISOLocaleContainer

ISOLocaleContainer

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO LocaleContainer

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOLocaleContainer`

Public fields

`description` description [1..1]

`locale` locale [1..1]

`date` date [1..*]

`responsibleParty` responsibleParty [1..*]

`localisedString` localisedString [1..*]

Methods

Public methods:

- `ISOLocaleContainer$new()`
- `ISOLocaleContainer$setDescription()`
- `ISOLocaleContainer$setLocale()`
- `ISOLocaleContainer$addDate()`
- `ISOLocaleContainer$delDate()`
- `ISOLocaleContainer$addResponsibleParty()`

- `ISOLocaleContainer$delResponsibleParty()`
- `ISOLocaleContainer$addLocalisedString()`
- `ISOLocaleContainer$delLocalisedString()`
- `ISOLocaleContainer$clone()`

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

Usage:

```
ISOLocaleContainer$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setDescription()`**:** Set description

Usage:

```
ISOLocaleContainer$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` list of localized texts. Default is `NULL`

Method `setLocale()`**:** Set locale

Usage:

```
ISOLocaleContainer$setLocale(locale)
```

Arguments:

`locale` object of class ISOLocale

Method `addDate()`**:** Adds date

Usage:

```
ISOLocaleContainer$addDate(date)
```

Arguments:

`date` object of class ISODate

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDate()`**:** Deletes date

Usage:

```
ISOLocaleContainer$delDate(date)
```

Arguments:

`date` object of class ISODate

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addResponsibleParty()`**:** Adds responsible party

Usage:

```
ISOLocaleContainer$addResponsibleParty(responsibleParty)
```

Arguments:

`responsibleParty` object of class ISOResponsibleParty

Returns: `TRUE` if added, `FALSE` otherwise

Method `delResponsibleParty()`**:** Deletes responsible party

Usage:

```
ISOLocaleContainer$delResponsibleParty(responsibleParty)
```

Arguments:

`responsibleParty` object of class ISOResponsibleParty

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addLocalisedString()`**:** Adds localised string

Usage:

```
ISOLocaleContainer$addLocalisedString(string)
```

Arguments:

`string` object of class character

Returns: `TRUE` if added, `FALSE` otherwise

Method `delLocalisedString()`**:** Deletes localised string

Usage:

```
ISOLocaleContainer$delLocalisedString(string)
```

Arguments:

`string` object of class character

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOLocalisedCharacterString` *ISOLocalisedCharacterString*

Description

ISOLocalisedCharacterString

ISOLocalisedCharacterString

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO LocalisedCharacterString

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOBaseCharacterString` -> `ISOLocalisedCharacterString`

Methods

Public methods:

- `ISOLocalisedCharacterString$new()`
- `ISOLocalisedCharacterString$clone()`

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

Usage:

```
ISOLocalisedCharacterString$new(xml = NULL, locale = NULL, value)
```

Arguments:

`xml` object of class XMLInternalNode-class

`locale` locale

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

Examples

```
str <- ISOLocalisedCharacterString$new(locale = "FR", value = "ma description")
str$encode()
```

`ISOLocalName` *ISOLocalName*

Description

ISOLocalName

ISOLocalName

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO LocalName

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::GMLCodeType` -> `geometa::ISOAbstractGene`
-> `ISOLocalName`

Public fields

`value` value

Methods

Public methods:

- `ISOLocalName$new()`
- `ISOLocalName$clone()`

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

Usage:

```
ISOLocalName$new(xml = NULL, value = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOMaintenanceFrequency` *ISOMaintenanceFrequency*

Description

ISOMaintenanceFrequency

ISOMaintenanceFrequency

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO MaintenanceFrequency

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOMaintenanceFrequency`

Methods

Public methods:

- `ISOMaintenanceFrequency$new()`
- `ISOMaintenanceFrequency$clone()`

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

Usage:

```
ISOMaintenanceFrequency$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOMaintenanceFrequency$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOMaintenanceFrequency$values(labels = TRUE)

#daily frequency
daily <- ISOMaintenanceFrequency$new(value = "daily")
```

`ISOMaintenanceInformation` *ISOMaintenanceInformation*

Description

ISOMaintenanceInformation

ISOMaintenanceInformation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO MaintenanceInformation

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> ISOMaintenanceInformation

Public fields

`maintenanceAndUpdateFrequency` maintenanceAndUpdateFrequency

Methods

Public methods:

- ISOMaintenanceInformation$new()
- ISOMaintenanceInformation$setMaintenanceFrequency()
- ISOMaintenanceInformation$clone()

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

Usage:

```
ISOMaintenanceInformation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setMaintenanceFrequency()`**:** Set maintenance frequency

Usage:

```
ISOMaintenanceInformation$setMaintenanceFrequency(frequency)
```

Arguments:

`frequency` frequency object of class ISOMaintenanceFrequency or any character among values returned by `ISOMaintenanceFrequency$values()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOMaintenanceInformation$new()
md$setMaintenanceFrequency("daily")
xml <- md$encode()
```

`ISOMeasure` *ISOMeasure*

Description

ISOMeasure

ISOMeasure

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Measure

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOMeasure`

Public fields

`value` value

`attrs` attrs

Methods

Public methods:

- `ISOMeasure$new()`
- `ISOMeasure$clone()`

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

Usage:

```
ISOMeasure$new(xml = NULL, value, uom, useUomURI = FALSE)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`uom` uom symbol of unit of measure used

`useUomURI` use uom URI. Default is `FALSE`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOMedium` *ISOMedium*

Description

ISOMedium

ISOMedium

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Citation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOMedium`

Public fields

`name` name

`density` density

`densityUnits` density units

`volumes` volumes

`mediumFormat` medium format

`mediumNote` medium note

Methods

Public methods:

- `ISOMedium$new()`
- `ISOMedium$setName()`
- `ISOMedium$addDensity()`
- `ISOMedium$delDensity()`
- `ISOMedium$setDensityUnits()`
- `ISOMedium$setVolumes()`

- `ISOMedium$addMediumFormat()`
- `ISOMedium$delMediumFormat()`
- `ISOMedium$setMediumNote()`
- `ISOMedium$clone()`

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

Usage:

```
ISOMedium$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setName()`**:** Set name

Usage:

```
ISOMedium$setName(name)
```

Arguments:

`name` name object of class ISOMediumName or character among values returned by `ISOMediumName$values()`

Method `addDensity()`**:** Adds density

Usage:

```
ISOMedium$addDensity(density)
```

Arguments:

`density` object of class numeric

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDensity()`**:** Deletes density

Usage:

```
ISOMedium$delDensity(density)
```

Arguments:

`density` object of class numeric

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setDensityUnits()`**:** Set density units

Usage:

```
ISOMedium$setDensityUnits(densityUnits)
```

Arguments:

`densityUnits` densityUnits

Method `setVolumes()`**:** Set volumes

Usage:

```
ISOMedium$setVolumes(volumes)
```

Arguments:

`volumes` object of class integer

Method `addMediumFormat()`**:** Adds medium format

Usage:

```
ISOMedium$addMediumFormat(mediumFormat)
```

Arguments:

`mediumFormat` object of class ISOMediumFormat or character among values returned by `ISOMediumFormat$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delMediumFormat()`**:** Deletes medium format

Usage:

```
ISOMedium$delMediumFormat(mediumFormat)
```

Arguments:

`mediumFormat` object of class ISOMediumFormat or character among values returned by `ISOMediumFormat$values()`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setMediumNote()`**:** Set medium note

Usage:

```
ISOMedium$setMediumNote(mediumNote, locales = NULL)
```

Arguments:

`mediumNote` medium note

`locales` list of localized notes. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOMedium$new()
md$setName("satellite")
md$addDensity(1.0)
md$setDensityUnits("string")
md$setVolumes(1L)
md$addMediumFormat("tar")
md$setMediumNote("some note")
xml <- md$encode()
```

ISOMediumFormat *ISOMediumFormat*

Description

ISOMediumFormat

ISOMediumFormat

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOMediumFormat

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOMediumFormat`

Methods

Public methods:

- `ISOMediumFormat$new()`
- `ISOMediumFormat$clone()`

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

Usage:

```
ISOMediumFormat$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOMediumFormat$values(labels = TRUE)

#MediumFormat
MediumFormat <- ISOMediumFormat$new(value = "tar")
```

ISOMediumName *ISOMediumName*

Description

ISOMediumName

ISOMediumName

Format

R6Class object.

Value

Object of R6Class for modelling an ISOMediumName

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISOCodeListValue -> ISOMediumName

Methods

Public methods:

- ISOMediumName$new()
- ISOMediumName$clone()

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

Usage:

```
ISOMediumName$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOMediumName$values(labels = TRUE)

#MediumName
MediumName <- ISOMediumName$new(value = "satellite")
```

`ISOMemberName` *ISOMemberName*

Description

ISOMemberName

ISOMemberName

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOMemberName

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOMemberName`

Public fields

`aName` name

`attributeType` attribute type

Methods

Public methods:

- `ISOMemberName$new()`
- `ISOMemberName$setName()`
- `ISOMemberName$setAttributeType()`
- `ISOMemberName$clone()`

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

Usage:

```
ISOMemberName$new(xml = NULL, aName = NULL, attributeType = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`aName` a name

`attributeType` attribute type

Method `setName()`**:** Set name

Usage:

```
ISOMemberName$setName(aName, locales = NULL)
```

Arguments:

`aName` name

`locales` list of localized texts. Default is `NULL`

Method `setAttributeType()`**:** Set attribute type

Usage:

```
ISOMemberName$setAttributeType(attributeType, locales = NULL)
```

Arguments:

`attributeType` attribute type

`locales` list of localized texts. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

`ISOMetadata` *ISOMetadata*

Description

ISOMetadata

ISOMetadata

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Metadata

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOMetadata`

Public fields

`fileIdentifier` fileIdentifier [0..1] : character

`language` language [0..1] : character

`characterSet` characterSet [0..1] : ISOCharacterSet = "utf8"

`parentIdentifier` parentIdentifier [0..1] : character

`hierarchyLevel` hierarchyLevel [0..*] : ISOHierarchyLevel = "dataset"

`hierarchyLevelName` hierarchyLevelName [0..*] : character

`contact` contact [1..*] : ISOResponsibleParty

`dateStamp` dateStamp : POSIXct/POSIXt

`metadataStandardName` metadataStandardName [0..1] : character

`metadataStandardVersion` metadataStandardVersion [0..1] : character

`dataSetURI` dataSetURI [0..1] : character

`locale` locale [0..*]: ISOLocale

`spatialRepresentationInfo` spatialRepresentationInfo [0..*]: ISOSpatialRepresentation

`referenceSystemInfo` referenceSystemInfo [0..*]: ISOReferenceSystem

`metadataExtensionInfo` metadataExtensionInfo [0..*]: ISOMetadataExtensionInformation

`identificationInfo` identificationInfo [1..*]: ISOIdentification

`contentInfo` contentInfo [0..*]

`distributionInfo` distributionInfo [0..1] : ISODistribution

`dataQualityInfo` dataQualityInfo [0..*]: ISODataQuality

`metadataMaintenance` metadataMaintenance [0..1]: ISOMaintenanceInformation

`portrayalCatalogueInfo` portrayalCatalogueInfo [0..*]

`applicationSchemaInformation` applicationSchemaInfo [0..*]

Methods

Public methods:

- `ISOMetadata$new()`
- `ISOMetadata$setFileIdentifier()`
- `ISOMetadata$setLanguage()`
- `ISOMetadata$setCharacterSet()`
- `ISOMetadata$setParentIdentifier()`
- `ISOMetadata$addHierarchyLevel()`
- `ISOMetadata$setHierarchyLevel()`
- `ISOMetadata$delHierarchyLevel()`
- `ISOMetadata$addHierarchyLevelName()`
- `ISOMetadata$delHierarchyLevelName()`
- `ISOMetadata$addContact()`
- `ISOMetadata$delContact()`
- `ISOMetadata$setDateStamp()`
- `ISOMetadata$setMetadataStandardName()`
- `ISOMetadata$setMetadataStandardVersion()`
- `ISOMetadata$setDataSetURI()`
- `ISOMetadata$addLocale()`
- `ISOMetadata$delLocale()`
- `ISOMetadata$addSpatialRepresentationInfo()`
- `ISOMetadata$setSpatialRepresentationInfo()`
- `ISOMetadata$delSpatialRepresentationInfo()`
- `ISOMetadata$addReferenceSystemInfo()`
- `ISOMetadata$setReferenceSystemInfo()`
- `ISOMetadata$delReferenceSystemInfo()`
- `ISOMetadata$addMetadataExtensionInfo()`
- `ISOMetadata$delMetadataExtensionInfo()`
- `ISOMetadata$addIdentificationInfo()`
- `ISOMetadata$setIdentificationInfo()`
- `ISOMetadata$delIdentificationInfo()`
- `ISOMetadata$setDistributionInfo()`
- `ISOMetadata$addDataQualityInfo()`
- `ISOMetadata$setDataQualityInfo()`
- `ISOMetadata$delDataQualityInfo()`
- `ISOMetadata$setMetadataMaintenance()`
- `ISOMetadata$addContentInfo()`
- `ISOMetadata$delContentInfo()`
- `ISOMetadata$clone()`

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

Usage:

`ISOMetadata$new(xml = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

Method `setFileIdentifier()`**:** Set file identifier

Usage:

`ISOMetadata$setFileIdentifier(fileIdentifier)`

Arguments:

`fileIdentifier` file identifier

Method `setLanguage()`**:** Set language

Usage:

`ISOMetadata$setLanguage(locale)`

Arguments:

`locale` object of class ISOLanguage or any character from values returned by `ISOLanguages$values()`

Method `setCharacterSet()`**:** Set charset

Usage:

`ISOMetadata$setCharacterSet(charset)`

Arguments:

`charset` object of class ISOCharacterSet or any character from values returned by `ISOCharacterSet$values()`

Method `setParentIdentifier()`**:** Set parent identifier

Usage:

`ISOMetadata$setParentIdentifier(parentIdentifier)`

Arguments:

`parentIdentifier` parent identifier

Method `addHierarchyLevel()`**:** Adds hierarchy level

Usage:

`ISOMetadata$addHierarchyLevel(level)`

Arguments:

`level` object of class ISOHierarchyLevel or any character from values returned by `ISOHierarchyLevel$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `setHierarchyLevel()`**:** Sets hierarchy level

Usage:

`ISOMetadata$setHierarchyLevel(level)`

Arguments:

`level` object of class ISOHierarchyLevel or any character from values returned by `ISOHierarchyLevel$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delHierarchyLevel()`**:** Deletes hierarchy level

Usage:

```
ISOMetadata$delHierarchyLevel(level)
```

Arguments:

`level` object of class ISOHierarchyLevel or any character from values returned by `ISOHierarchyLevel$values()`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addHierarchyLevelName()`**:** Adds hierarchy level name

Usage:

```
ISOMetadata$addHierarchyLevelName(levelName)
```

Arguments:

`levelName` object of class character

Returns: `TRUE` if added, `FALSE` otherwise

Method `delHierarchyLevelName()`**:** Deletes hierarchy level name

Usage:

```
ISOMetadata$delHierarchyLevelName(levelName)
```

Arguments:

`levelName` object of class character

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addContact()`**:** Adds contact

Usage:

```
ISOMetadata$addContact(contact)
```

Arguments:

`contact` object of class ISOResponsibleParty

Returns: `TRUE` if added, `FALSE` otherwise

Method `delContact()`**:** Deletes contact

Usage:

```
ISOMetadata$delContact(contact)
```

Arguments:

`contact` object of class ISOResponsibleParty

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setDateStamp()`**:** Set date stamp

Usage:

```
ISOMetadata$setDateStamp(date)
```

Arguments:

`date` date

Method `setMetadataStandardName()`: Set metadata standard name

Usage:

```
ISOMetadata$setMetadataStandardName(name)
```

Arguments:

`name` name

Method `setMetadataStandardVersion()`: Set metadata standard version

Usage:

```
ISOMetadata$setMetadataStandardVersion(version)
```

Arguments:

`version` version

Method `setDataSetURI()`: Set dataset URI

Usage:

```
ISOMetadata$setDataSetURI(dataSetURI)
```

Arguments:

`dataSetURI` dataset URI

Method `addLocale()`: Adds locale

Usage:

```
ISOMetadata$addLocale(locale)
```

Arguments:

`locale` object of class ISOLocale

Returns: `TRUE` if added, `FALSE` otherwise

Method `delLocale()`: Deletes locale

Usage:

```
ISOMetadata$delLocale(locale)
```

Arguments:

`locale` object of class ISOLocale

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addSpatialRepresentationInfo()`: Adds spatial representation info

Usage:

```
ISOMetadata$addSpatialRepresentationInfo(spatialRepresentationInfo)
```

Arguments:

`spatialRepresentationInfo` object of class ISOSpatialRepresentation

Returns: `TRUE` if added, `FALSE` otherwise

Method `setSpatialRepresentationInfo()`: Sets spatial representation info

Usage:

```
ISOMetadata$setSpatialRepresentationInfo(spatialRepresentationInfo)
```

Arguments:

`spatialRepresentationInfo` object of class ISOSpatialRepresentation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSpatialRepresentationInfo()`**:** Deletes spatial representation info

Usage:

```
ISOMetadata$delSpatialRepresentationInfo(spatialRepresentationInfo)
```

Arguments:

`spatialRepresentationInfo` object of class ISOSpatialRepresentation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addReferenceSystemInfo()`**:** Adds reference system info

Usage:

```
ISOMetadata$addReferenceSystemInfo(referenceSystemInfo)
```

Arguments:

`referenceSystemInfo` object of class ISOReferenceSystem

Returns: `TRUE` if added, `FALSE` otherwise

Method `setReferenceSystemInfo()`**:** Sets reference system info

Usage:

```
ISOMetadata$setReferenceSystemInfo(referenceSystemInfo)
```

Arguments:

`referenceSystemInfo` object of class ISOReferenceSystem

Returns: `TRUE` if added, `FALSE` otherwise

Method `delReferenceSystemInfo()`**:** Deletes reference system info

Usage:

```
ISOMetadata$delReferenceSystemInfo(referenceSystemInfo)
```

Arguments:

`referenceSystemInfo` object of class ISOReferenceSystem

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addMetadataExtensionInfo()`**:** Adds metadata extension info

Usage:

```
ISOMetadata$addMetadataExtensionInfo(metadataExtensionInfo)
```

Arguments:

`metadataExtensionInfo` object of class ISOMetadataExtensionInformation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delMetadataExtensionInfo()`**:** Deletes metadata extension info

Usage:

`ISOMetadata$delMetadataExtensionInfo(metadataExtensionInfo)`

Arguments:

`metadataExtensionInfo` object of class ISOMetadataExtensionInformation

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addIdentificationInfo()`**:** Adds metadata extension info

Usage:

`ISOMetadata$addIdentificationInfo(identificationInfo)`

Arguments:

`identificationInfo` object of class inheriting ISOIdentification

Returns: `TRUE` if added, `FALSE` otherwise

Method `setIdentificationInfo()`**:** Sets metadata extension info

Usage:

`ISOMetadata$setIdentificationInfo(identificationInfo)`

Arguments:

`identificationInfo` object of class inheriting ISOIdentification

Returns: `TRUE` if added, `FALSE` otherwise

Method `delIdentificationInfo()`**:** Deletes metadata extension info

Usage:

`ISOMetadata$delIdentificationInfo(identificationInfo)`

Arguments:

`identificationInfo` object of class inheriting ISOIdentification

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setDistributionInfo()`**:** Sets metadata extension info

Usage:

`ISOMetadata$setDistributionInfo(distributionInfo)`

Arguments:

`distributionInfo` object of class ISODistribution

Returns: `TRUE` if set, `FALSE` otherwise

Method `addDataQualityInfo()`**:** Adds data quality info

Usage:

`ISOMetadata$addDataQualityInfo(dataQualityInfo)`

Arguments:

`dataQualityInfo` object of class ISODataQuality

Returns: `TRUE` if added, `FALSE` otherwise

Method `setDataQualityInfo()`**:** Sets data quality info

Usage:

`ISOMetadata$setDataQualityInfo(dataQualityInfo)`

Arguments:

`dataQualityInfo` object of class ISODataQuality

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDataQualityInfo()`**:** Deletes data quality info

Usage:

`ISOMetadata$delDataQualityInfo(dataQualityInfo)`

Arguments:

`dataQualityInfo` object of class ISODataQuality

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setMetadataMaintenance()`**:** Sets metadata maintenance

Usage:

`ISOMetadata$setMetadataMaintenance(metadataMaintenance)`

Arguments:

`metadataMaintenance` object of class ISOMaintenanceInformation

Returns: `TRUE` if added, `FALSE` otherwise

Method `addContentInfo()`**:** Adds content information

Usage:

`ISOMetadata$addContentInfo(contentInfo)`

Arguments:

`contentInfo` object of class inheriting ISOContentInformation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delContentInfo()`**:** Deletes content information

Usage:

`ISOMetadata$delContentInfo(contentInfo)`

Arguments:

`contentInfo` object of class inheriting ISOContentInformation

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOMetadata$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
 #example 1 - WRITE: Create an ISO metadata and encode it as XML
 #######################################################
 md = ISOMetadata$new()
 md$setFileIdentifier("my-metadata-identifier")
 md$setParentIdentifier("my-parent-metadata-identifier")
 md$setCharacterSet("utf8")
 md$setLanguage("eng")
 md$setDateStamp(ISOdate(2015, 1, 1, 1))
 md$setMetadataStandardName("ISO 19115:2003/19139")
 md$setMetadataStandardVersion("1.0")
 md$setDataSetURI("my-dataset-identifier")

 #add 3 contacts
 for(i in 1:3){
   rp <- ISOResponsibleParty$new()
   rp$setIndividualName(paste0("someone",i))
   rp$setOrganisationName("somewhere")
   rp$setPositionName(paste0("someposition",i))
   rp$setRole("pointOfContact")
   contact <- ISOContact$new()
   phone <- ISOTelephone$new()
   phone$setVoice(paste0("myphonenumber",i))
   phone$setFacsimile(paste0("myfacsimile",i))
   contact$setPhone(phone)
   address <- ISOAddress$new()
   address$setDeliveryPoint("theaddress")
   address$setCity("thecity")
   address$setPostalCode("111")
   address$setCountry("France")
   address$setEmail("someone@theorg.org")
   contact$setAddress(address)
   res <- ISOOnlineResource$new()
   res$setLinkage("http://somelink")
   res$setName("someresourcename")
   contact$setOnlineResource(res)
   rp$setContactInfo(contact)
   md$addContact(rp)
}

#VectorSpatialRepresentation
vsr <- ISOVectorSpatialRepresentation$new()
vsr$setTopologyLevel("geometryOnly")
geomObject <- ISOGeometricObjects$new()
geomObject$setGeometricObjectType("surface")
geomObject$setGeometricObjectCount(5L)
vsr$addGeometricObjects(geomObject)
md$addSpatialRepresentationInfo(vsr)
```

```
#ReferenceSystem
rs <- ISOReferenceSystem$new()
rsId <- ISOReferenceIdentifier$new(code = "4326", codeSpace = "EPSG")
rs$setReferenceSystemIdentifier(rsId)
md$addReferenceSystemInfo(rs)

#data identification
ident <- ISODataIdentification$new()
ident$setAbstract("abstract")
ident$setPurpose("purpose")
ident$addCredit("credit1")
ident$addCredit("credit2")
ident$addCredit("credit3")
ident$addStatus("completed")
ident$addLanguage("eng")
ident$addCharacterSet("utf8")
ident$addTopicCategory("biota")
ident$addTopicCategory("oceans")

#adding a point of contact
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
rp$setOrganisationName("somewhere")
rp$setPositionName("someposition")
rp$setRole("pointOfContact")
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://somelink")
res$setName("somename")
contact$setOnlineResource(res)
rp$setContactInfo(contact)
ident$addPointOfContact(rp)

#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
```

```
ct$setEditionDate(as.Date(ISOdate(2015, 1, 1, 1)))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp)
ident$setCitation(ct)

#graphic overview
go1 <- ISOBrowseGraphic$new(
  fileName = "http://wwww.somefile.org/png1",
  fileDescription = "Map Overview 1",
  fileType = "image/png"
)
go2 <- ISOBrowseGraphic$new(
  fileName = "http://www.somefile.org/png2",
  fileDescription = "Map Overview 2",
  fileType = "image/png"
)
ident$addGraphicOverview(go1)
ident$addGraphicOverview(go2)

#maintenance information
mi <- ISOMaintenanceInformation$new()
mi$setMaintenanceFrequency("daily")
ident$addResourceMaintenance(mi)

#adding legal constraints
lc <- ISOLegalConstraints$new()
lc$addUseLimitation("limitation1")
lc$addUseLimitation("limitation2")
lc$addUseLimitation("limitation3")
lc$addAccessConstraint("copyright")
lc$addAccessConstraint("license")
lc$addUseConstraint("copyright")
lc$addUseConstraint("license")
ident$addResourceConstraints(lc)

#adding security constraints
sc <- ISOSecurityConstraints$new()
sc$setClassification("secret")
sc$setUserNote("ultra secret")
sc$setClassificationSystem("no classification in particular")
sc$setHandlingDescription("description")
ident$addResourceConstraints(sc)

#adding extent
extent <- ISOExtent$new()
bbox <- ISOGeographicBoundingBox$new(minx = -180, miny = -90, maxx = 180, maxy = 90)
extent$addGeographicElement(bbox)
ident$addExtent(extent)

#add keywords
kwds <- ISOKeywords$new()
kwds$addKeyword("keyword1")
```

```
kwds$addKeyword("keyword2")
kwds$setKeywordType("theme")
th <- ISOCitation$new()
th$setTitle("General")
th$addDate(d)
kwds$setThesaurusName(th)
ident$addKeywords(kwds)

#add an INSPIRE spatial data theme?
inspire_kwd <- ISOKeywords$new()
anc1 <- ISOAnchor$new(
  name = "Environmental monitoring facilities",
  href = "http://inspire.ec.europa.eu/theme/ef"
)
inspire_kwd$addKeyword(anc1)
inspire_kwd$setKeywordType("theme")
th <- ISOCitation$new()
th$setTitle(
  ISOAnchor$new(
    name = "GEMET - INSPIRE themes, version 1.0",
    href="http://www.eionet.europa.eu/gemet/inspire_themes"
  )
)
inspire_date <- ISODate$new()
inspire_date$setDate(as.Date("2008-06-01"))
inspire_date$setDateType("publication")
th$addDate(inspire_date)
inspire_kwd$setThesaurusName(th)
ident$addKeywords(inspire_kwd)

#supplementalInformation
ident$setSupplementalInformation("some additional information")

#spatial representation type
ident$addSpatialRepresentationType("vector")

md$addIdentificationInfo(ident)

#Distribution
distrib <- ISODistribution$new()
dto <- ISODigitalTransferOptions$new()
for(i in 1:3){
  or <- ISOOnlineResource$new()
  or$setLinkage(paste0("http://somelink",i))
  or$setName(paste0("name",i))
  or$setDescription(paste0("description",i))
  or$setProtocol("WWW:LINK-1.0-http--link")
  dto$addOnlineResource(or)
}
distrib$setDigitalTransferOptions(dto)
md$setDistributionInfo(distrib)

#create dataQuality object with a 'dataset' scope
```

```
dq <- ISODataQuality$new()
scope <- ISOScope$new()
scope$setLevel("dataset")
dq$setScope(scope)

#add data quality reports...

#add a report the data quality
dc <- ISODomainConsistency$new()
result <- ISOConformanceResult$new()
spec <- ISOCitation$new()
spec$setTitle("Data Quality check")
spec$addAlternateTitle("This is is some data quality check report")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dc$addResult(result)
dq$addReport(dc)

#add INSPIRE reports?
#INSPIRE - interoperability of spatial data sets and services
dc_inspire1 <- ISODomainConsistency$new()
cr_inspire1 <- ISOConformanceResult$new()
cr_inspire_spec1 <- ISOCitation$new()
cr_title1 <- paste(
 "Commission Regulation (EU) No 1089/2010 of 23 November 2010 implementing Directive 2007/2/EC",
 "of the European Parliament and of the Council as regards interoperability of spatial data",
   "sets and services"
)
cr_inspire_spec1$setTitle(cr_title1)
cr_inspire1$setExplanation("See the referenced specification")
cr_inspire_date1 <- ISODate$new()
cr_inspire_date1$setDate(ISOdate(2010,12,8))
cr_inspire_date1$setDateType("publication")
cr_inspire_spec1$addDate(cr_inspire_date1)
cr_inspire1$setSpecification(cr_inspire_spec1)
cr_inspire1$setPass(TRUE)
dc_inspire1$addResult(cr_inspire1)
dq$addReport(dc_inspire1)
#INSPIRE - metadata
dc_inspire2 <- ISODomainConsistency$new()
cr_inspire2 <- ISOConformanceResult$new()
cr_inspire_spec2 <- ISOCitation$new()
cr_title2 <- paste(
 "COMMISSION REGULATION (EC) No 1205/2008 of 3 December 2008 implementing Directive 2007/2/EC",
   "of the European Parliament and of the Council as regards metadata"
)
cr_inspire_spec2$setTitle(cr_title2)
cr_inspire2$setExplanation("See the referenced specification")
```

```
cr_inspire_date2 <- ISODate$new()
cr_inspire_date2$setDate(ISOdate(2008,12,4))
cr_inspire_date2$setDateType("publication")
cr_inspire_spec2$addDate(cr_inspire_date2)
cr_inspire2$setSpecification(cr_inspire_spec2)
cr_inspire2$setPass(TRUE)
dc_inspire2$addResult(cr_inspire2)
dq$addReport(dc_inspire2)

#add lineage
lineage <- ISOLineage$new()
lineage$setStatement("statement")
dq$setLineage(lineage)

md$addDataQualityInfo(dq)

#Content Information
#-------------------------
#add a feature catalogue description
fcd <- ISOFeatureCatalogueDescription$new()
fcd$setComplianceCode(FALSE)
fcd$addLanguage("eng")
fcd$setIncludedWithDataset(FALSE)
cit = ISOCitation$new()
cit$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
cit$addDate(d)
cit$setEdition("1.0")
cit$setEditionDate(as.Date(ISOdate(2015, 1, 1, 1)))
contact = ISOContact$new()
fcLink <- ISOOnlineResource$new()
fcLink$setLinkage("http://somelink/featurecatalogue")
contact$setOnlineResource(fcLink)
rp = ISOResponsibleParty$new()
rp$setRole("publisher")
rp$setContactInfo(contact)
cit$addCitedResponsibleParty(rp)
fcd$addFeatureCatalogueCitation(cit)
md$addContentInfo(fcd)

#XML representation of the ISOMetadata
xml <- md$encode()

#example 2 - READ: Create an ISO metadata reading from XML
######################################################

require(XML)
xmlfile <- system.file("extdata/examples", "metadata.xml", package = "geometa")
xml <- xmlParse(xmlfile)
md <- ISOMetadata$new(xml = xml)
```

ISOMetadataExtensionInformation
ISOMetadataExtensionInformation

Description

ISOMetadataExtensionInformation

ISOMetadataExtensionInformation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO MetadataExtensionInformation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOMetadataExtensionInformation`

Public fields

`extensionOnLineResource` extensionOnLineResource [0..1]: ISOOnlineResource

`extendedElementInformation` extendedElementInformation [0..*]: ISOExtendedElementInformation

Methods

Public methods:

- `ISOMetadataExtensionInformation$new()`
- `ISOMetadataExtensionInformation$setOnlineResource()`
- `ISOMetadataExtensionInformation$addElement()`
- `ISOMetadataExtensionInformation$delElement()`
- `ISOMetadataExtensionInformation$clone()`

Method `new()`: Initializes object

Usage:

```
ISOMetadataExtensionInformation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setOnlineResource()`: Set online resource

Usage:

```
ISOMetadataExtensionInformation$setOnlineResource(onlineResource)
```

Arguments:

`onlineResource` object of class ISOOnlineResource

Method `addElement()`**:** Adds element

Usage:

```
ISOMetadataExtensionInformation$addElement(element)
```

Arguments:

`element` object of class inheriting ISOExtendedElementInformation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delElement()`**:** Deletes element

Usage:

```
ISOMetadataExtensionInformation$delElement(element)
```

Arguments:

`element` object of class inheriting ISOExtendedElementInformation

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create an extended element information
elem <- ISOExtendedElementInformation$new()
elem$setName("name")
elem$setShortName("shortName")
elem$setDomainCode(1L)
elem$setDefinition("some definition")
elem$setObligation("mandatory")
elem$setCondition("no condition")
elem$setDatatype("characterString")
elem$setMaximumOccurrence("string")
elem$setDomainValue("value")
```

```
elem$addParentEntity("none")
elem$setRule("rule")
elem$addRationale("rationale")
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
rp$setOrganisationName("somewhere")
rp$setPositionName("someposition")
rp$setRole("pointOfContact")
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact$setOnlineResource(res)
rp$setContactInfo(contact)
elem$addSource(rp)

md <- ISOMetadataExtensionInformation$new()
md$addElement(elem)

xml <- md$encode()
```

ISOMetadataNamespace *ISOMetadataNamespace*

Description

ISOMetadataNamespace

ISOMetadataNamespace

Format

R6Class object.

Value

Object of R6Class for modelling an ISO Metadata Namespace

Public fields

`id` id

`uri` uri

Methods

Public methods:

- `ISOMetadataNamespace$new()`
- `ISOMetadataNamespace$getDefinition()`
- `ISOMetadataNamespace$clone()`

Method `new()`**:** Initializes namespace object

Usage:

```
ISOMetadataNamespace$new(id, uri)
```

Arguments:

`id` id

`uri` uri

Method `getDefinition()`**:** Get definition

Usage:

```
ISOMetadataNamespace$getDefinition()
```

Returns: an object of class list

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

ISO class used internally by geometa for specifying XML namespaces

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

`ISOMetaIdentifier` *ISOMetaIdentifier*

Description

ISOMetaIdentifier

ISOMetaIdentifier

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO MetaIdentifier

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOIdentifier` -> `ISOMetaIdentifier`

Methods

Public methods:

- `ISOMetaIdentifier$new()`
- `ISOMetaIdentifier$clone()`

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

Usage:

```
ISOMetaIdentifier$new(xml = NULL, code)
```

Arguments:

`xml` object of class XMLInternalNode-class

`code` code

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOMetaIdentifier$new(code = "identifier")
xml <- md$encode()
```

`ISOMimeFileType` *ISOMimeFileType*

Description

ISOMimeFileType

ISOMimeFileType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO MimeFileType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOMimeFileType`

Methods

Public methods:

- `ISOMimeFileType$new()`
- `ISOMimeFileType$setName()`
- `ISOMimeFileType$setType()`
- `ISOMimeFileType$clone()`

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

Usage:

```
ISOMimeFileType$new(xml = NULL, type = NULL, name = NULL)
```

Arguments:

`xml`	object of class XMLInternalNode-class
`type`	type
`name`	name

Method `setName()`: Set name

Usage:

`ISOMimeFileType$setName(name)`

Arguments:

`name` name

Method `setType()`: Set type

Usage:

`ISOMimeFileType$setType(type)`

Arguments:

`type` type

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

Usage:

`ISOMimeFileType$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19139:2007 Geographic information – XML

Examples

```
md <- ISOMimeFileType$new(type = "somemimetype", name = "Mime type name")
xml <- md$encode()
```

`ISOMultiplicity` *ISOMultiplicity*

Description

ISOMultiplicity

ISOMultiplicity

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOMultiplicity

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOMultiplicity`

Public fields

`range` range

Methods

Public methods:

- `ISOMultiplicity$new()`
- `ISOMultiplicity$clone()`

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

Usage:

```
ISOMultiplicity$new(xml = NULL, lower, upper)
```

Arguments:

`xml` object of class XMLInternalNode-class

`lower` lower

`upper` upper

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

Examples

```
md <- ISOMultiplicity$new(lower = 1, upper = Inf)
xml <- md$encode()
```

`ISOMultiplicityRange` *ISOMultiplicityRange*

Description

ISOMultiplicityRange

ISOMultiplicityRange

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO MultiplicityRange

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOMultiplicityRange`

Public fields

`lower` lower

`upper` upper

Methods

Public methods:

- `ISOMultiplicityRange$new()`
- `ISOMultiplicityRange$clone()`

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

Usage:

```
ISOMultiplicityRange$new(xml = NULL, lower, upper)
```

Arguments:

`xml` object of class XMLInternalNode-class

`lower` lower

`upper` upper

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

Examples

```
md <- ISOMultiplicityRange$new(lower = 1, upper = Inf)
xml <- md$encode()
```

ISONonQuantitativeAttributeAccuracy
ISONonQuantitativeAttributeAccuracy

Description

ISONonQuantitativeAttributeAccuracy

ISONonQuantitativeAttributeAccuracy

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISONonQuantitativeAttributeAccuracy

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `geometa::ISOAbstractThematicAccuracy` -> `ISONonQuantitativeAttributeAccuracy`

Methods

Public methods:

- `ISONonQuantitativeAttributeAccuracy$clone()`

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

Usage:

`ISONonQuantitativeAttributeAccuracy$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISONonQuantitativeAttributeAccuracy$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

`ISOObligation` *ISOObligation*

Description

ISOObligation

ISOObligation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Obligation

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISOCodeListValue -> ISOObligation

Methods

Public methods:

- ISOObligation$new()
- ISOObligation$clone()

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

Usage:

```
ISOObligation$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOObligation$values(labels = TRUE)

#mandatory value
mandatory <- ISOObligation$new(value = "mandatory")
```

`ISOOnLineFunction` *ISOOnLineFunction*

Description

ISOOnLineFunction

ISOOnLineFunction

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO OnLineFunction

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOOnLineFunction`

Methods

Public methods:

- `ISOOnLineFunction$new()`
- `ISOOnLineFunction$clone()`

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

Usage:

```
ISOOnLineFunction$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOOnLineFunction$values(labels = TRUE)

#example
download <- ISOOnLineFunction$new(value = "download")
```

`ISOOnlineResource` *ISOOnlineResource*

Description

ISOOnlineResource

ISOOnlineResource

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Online Resource

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOOnlineResource`

Public fields

`linkage` linkage

`protocol` protocol

`name` name

`description` description

`function` function

Methods

Public methods:

- `ISOOnlineResource$new()`
- `ISOOnlineResource$setLinkage()`
- `ISOOnlineResource$setName()`
- `ISOOnlineResource$setProtocol()`
- `ISOOnlineResource$setDescription()`
- `ISOOnlineResource$setOnLineFunction()`
- `ISOOnlineResource$clone()`

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

Usage:

```
ISOOnlineResource$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setLinkage()`**:** Set linkage

Usage:

```
ISOOnlineResource$setLinkage(linkage)
```

Arguments:

`linkage` linkage object of class ISOURL or character

Method `setName()`**:** Set name

Usage:

```
ISOOnlineResource$setName(name, locales = NULL)
```

Arguments:

`name` name

`locales` list of localized texts. Default is `NULL`

Method `setProtocol()`**:** Set protocol

Usage:

```
ISOOnlineResource$setProtocol(protocol, locales = NULL)
```

Arguments:

`protocol` protocol

`locales` list of localized texts. Default is `NULL`

Method `setDescription()`**:** Set description

Usage:

```
ISOOnlineResource$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` list of localized texts. Default is `NULL`

Method `setOnLineFunction()`**:** Set online function

Usage:

```
ISOOnlineResource$setOnLineFunction(onLineFunction)
```

Arguments:

`onLineFunction` object of class ISOOnLineFunction or any character among values returned by `ISOOnLineFunction$values()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOOnlineResource$new()
md$setLinkage("http://somelink")
md$setName("name")
md$setDescription("description")
md$setProtocol("protocol")
md$setOnLineFunction("download")
xml <- md$encode()
```

`ISOOperationMetadata` *ISOOperationMetadata*

Description

ISOOperationMetadata

ISOOperationMetadata

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOOperationMetadata

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOOperationMetadata`

Public fields

`operationName` operationName [1..1]: character

`DCP` DCP [1..*]: ISODCPList

`operationDescription` operationDescription [0..1]: character

`invocationName` invocationName [0..1]: character

`parameters` parameters [0..*]: ISOParameter

`connectPoint` connectPoint [1..*]: ISOOnlineResource

`dependsOn` dependsOn [0..*]: ISOOperationMetadata

Methods

Public methods:

- `ISOOperationMetadata$new()`
- `ISOOperationMetadata$setOperationName()`
- `ISOOperationMetadata$addDCP()`
- `ISOOperationMetadata$delDCP()`
- `ISOOperationMetadata$setOperationDescription()`
- `ISOOperationMetadata$setInvocationName()`
- `ISOOperationMetadata$addParameter()`
- `ISOOperationMetadata$delParameter()`
- `ISOOperationMetadata$addConnectPoint()`
- `ISOOperationMetadata$delConnectPoint()`
- `ISOOperationMetadata$addDependentOperationMetadata()`
- `ISOOperationMetadata$delDependentOperationMetadata()`
- `ISOOperationMetadata$clone()`

Method `new()`: Initializes object

Usage:

```
ISOOperationMetadata$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setOperationName()`: Set operation name

Usage:

```
ISOOperationMetadata$setOperationName(operationName, locales = NULL)
```

Arguments:

`operationName` operation name

`locales` list of localized texts. Default is `NULL`

Method `addDCP()`: Adds DCP

Usage:

```
ISOOperationMetadata$addDCP(dcp)
```

Arguments:

`dcp` object of class ISODCPList or any character among values returned by `ISODCPList$values()`

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDCP()`: Deletes DCP

Usage:

```
ISOOperationMetadata$delDCP(dcp)
```

Arguments:

`dcp` object of class ISODCPList or any character among values returned by `ISODCPList$values()`

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setOperationDescription()`: Set operation description

Usage:

```
ISOOperationMetadata$setOperationDescription(
  operationDescription,
  locales = NULL
)
```

Arguments:

`operationDescription` operation description

`locales` list of localized texts. Default is `NULL`

Method `setInvocationName()`: Set invocation name

Usage:

```
ISOOperationMetadata$setInvocationName(invocationName, locales = NULL)
```

Arguments:

`invocationName` invocation name

`locales` list of localized texts. Default is `NULL`

Method `addParameter()`: Adds parameter

Usage:

```
ISOOperationMetadata$addParameter(parameter)
```

Arguments:

`parameter` object of class ISOParameter

Returns: `TRUE` if added, `FALSE` otherwise

Method `delParameter()`: Deletes parameter

Usage:

```
ISOOperationMetadata$delParameter(parameter)
```

Arguments:

`parameter` object of class ISOParameter

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addConnectPoint():` Adds connection point

Usage:

```
ISOOperationMetadata$addConnectPoint(connectPoint)
```

Arguments:

`connectPoint` object of class ISOOnlineResource

Returns: `TRUE` if added, `FALSE` otherwise

Method `delConnectPoint():` Deletes connection point

Usage:

```
ISOOperationMetadata$delConnectPoint(connectPoint)
```

Arguments:

`connectPoint` object of class ISOOnlineResource

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addDependentOperationMetadata():` Adds operation metadata

Usage:

```
ISOOperationMetadata$addDependentOperationMetadata(operationMetadata)
```

Arguments:

`operationMetadata` object of class ISOOperationMetadata

Returns: `TRUE` if added, `FALSE` otherwise

Method `delDependentOperationMetadata():` Deletes operation metadata

Usage:

```
ISOOperationMetadata$delDependentOperationMetadata(operationMetadata)
```

Arguments:

`operationMetadata` object of class ISOOperationMetadata

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19119:2005 - Geographic information – Services

Examples

```
md <- ISOOperationMetadata$new()
xml <- md$encode()
```

`ISOOtherAggregate` *ISOOtherAggregate*

Description

ISOOtherAggregate

ISOOtherAggregate

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOOtherAggregate

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractAggregate` -> `ISOOtherAggregate`

Methods

Public methods:

- `ISOOtherAggregate$new()`
- `ISOOtherAggregate$clone()`

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

Usage:

`ISOOtherAggregate$new(xml = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

`ISOOtherAggregate$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOParameter` *ISOParameter*

Description

ISOParameter

ISOParameter

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOParameter

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOParameter`

Public fields

`name` name [1..1]: character

`direction` direction [0..1]: ISOParameterDirection

`description` description [0..1]: character

`optionality` optionality [1..1]: character

`repeatability` repeatability [1..1]: logical

`valueType` valueType [1..1]: ISOTypeName

Methods

Public methods:

- `ISOParameter$new()`
- `ISOParameter$setName()`
- `ISOParameter$setDirection()`
- `ISOParameter$setDescription()`
- `ISOParameter$setOptionality()`
- `ISOParameter$setRepeatability()`

- `ISOParameter$setValueType()`
- `ISOParameter$clone()`

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

Usage:

```
ISOParameter$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setName()`**:** Set name

Usage:

```
ISOParameter$setName(name, attributeType, locales = NULL)
```

Arguments:

`name` name

`attributeType` attribute type

`locales` list of localized texts. Default is `NULL`

Method `setDirection()`**:** Set direction

Usage:

```
ISOParameter$setDirection(direction)
```

Arguments:

`direction` object of class ISOParameterDirection or character among values returned by `ISOParameterDirection$val`

Method `setDescription()`**:** Set description

Usage:

```
ISOParameter$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` list of localized texts. Default is `NULL`

Method `setOptionality()`**:** Set optionality

Usage:

```
ISOParameter$setOptionality(optional)
```

Arguments:

`optional` object of class logical

Method `setRepeatability()`**:** Set repeatability

Usage:

```
ISOParameter$setRepeatability(repeatable)
```

Arguments:

`repeatable` object of class logical

Method `setValueType()`**:** Set value type

Usage:

`ISOParameter$setValueType(valueType, locales = NULL)`

Arguments:

`valueType` object of class ISOTypeName or character

`locales` list of localized texts. Default is `NULL`

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

Usage:

`ISOParameter$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19119:2005 - Geographic information – Services

Examples

```
md <- ISOParameter$new()
md$setName("name", "attType")
md$setDirection("in")
md$setDescription("description")
md$setOptionality(FALSE)
md$setRepeatability(FALSE)
md$setValueType("CharacterString")
xml <- md$encode()
```

`ISOParameterDirection` *ISOParameterDirection*

Description

ISOParameterDirection

ISOParameterDirection

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOParameterDirection

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOParameterDirection`

Methods

Public methods:

- `ISOParameterDirection$new()`
- `ISOParameterDirection$clone()`

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

Usage:

```
ISOParameterDirection$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19119:2005 - Geographic information – Services

Examples

```
#possible values
values <- ISOParameterDirection$values(labels = TRUE)

#paramDir
paramDir <- ISOParameterDirection$new(value = "in")
```

`ISOPixelOrientation` *ISOPixelOrientation*

Description

ISOPixelOrientation

ISOPixelOrientation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOPixelOrientation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOPixelOrientation`

Methods

Public methods:

- `ISOPixelOrientation$new()`
- `ISOPixelOrientation$clone()`

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

Usage:

```
ISOPixelOrientation$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOPixelOrientation$values(labels = TRUE)

#PixelOrientation
PixelOrientation <- ISOPixelOrientation$new(value = "center")
```

`ISOPlatform` *ISOPlatform*

Description

ISOPlatform

ISOPlatform

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOPlatform

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractAggregate` -> `geometa::ISOSeries` -> `ISOPlatform`

Methods

Public methods:

- `ISOPlatform$new()`
- `ISOPlatform$clone()`

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

Usage:

`ISOPlatform$new(xml = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOPortrayalCatalogueReference` *ISOPortrayalCatalogueReference*

Description

ISOPortrayalCatalogueReference

ISOPortrayalCatalogueReference

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOPortrayalCatalogueReference

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOPortrayalCatalogueReference`

Public fields

`portrayalCatalogueCitation` portrayalCatalogueCitation [1..*]

Methods

Public methods:

- `ISOPortrayalCatalogueReference$new()`
- `ISOPortrayalCatalogueReference$addCitation()`
- `ISOPortrayalCatalogueReference$delCitation()`
- `ISOPortrayalCatalogueReference$clone()`

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

Usage:

`ISOPortrayalCatalogueReference$new(xml = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

Method `addCitation()`**:** Adds citation

Usage:

`ISOPortrayalCatalogueReference$addCitation(citation)`

Arguments:

`citation` object of class ISOCitation

Returns: `TRUE` if added, `FALSE` otherwise

Method `delCitation()`**:** Deletes citation

Usage:

`ISOPortrayalCatalogueReference$delCitation(citation)`

Arguments:

`citation` object of class ISOCitation

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOPortrayalCatalogueReference$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOPortrayalCatalogueReference$new()
#citation
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
rp$setOrganisationName("somewhere")
rp$setPositionName("someposition")
rp$setRole("pointOfContact")
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
```

```
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://somelink")
res$setName("somename")
contact$setOnlineResource(res)
rp$setContactInfo(contact)
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(as.Date(ISOdate(2015, 1, 1, 1)))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp)
md$addCitation(ct)

xml <- md$encode()
```

`ISOPresentationForm` *ISOPresentationForm*

Description

ISOPresentationForm

ISOPresentationForm

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO PresentationForm

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOPresentationForm`

Methods

Public methods:

- `ISOPresentationForm$new()`
- `ISOPresentationForm$clone()`

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

Usage:

`ISOPresentationForm$new(xml = NULL, value, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOPresentationForm$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOPresentationForm$values(labels = TRUE)

#mapDigital type
map <- ISOPresentationForm$new(value = "mapDigital")
```

`ISOProcessStep` *ISOProcessStep*

Description

ISOProcessStep

ISOProcessStep

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO ProcessStep

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOProcessStep`

Public fields

`description` description: character

`rationale` rationale [0..1]: character

`dateTime` dateTime [0..1]: ISOBaseDateTime or POSIXct/POSIXt

`processor` processor [0..*]: ISOResponsibleParty

`source` source [0..*]: ISOSource

Methods

Public methods:

- `ISOProcessStep$new()`
- `ISOProcessStep$setDescription()`
- `ISOProcessStep$setRationale()`
- `ISOProcessStep$setDateTime()`
- `ISOProcessStep$addProcessor()`
- `ISOProcessStep$delProcessor()`
- `ISOProcessStep$addSource()`
- `ISOProcessStep$delSource()`
- `ISOProcessStep$clone()`

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

Usage:

```
ISOProcessStep$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setDescription()`**:** Set description

Usage:

```
ISOProcessStep$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` list of localized texts. Default is `NULL`

Method `setRationale()`: Set rationale

Usage:

```
ISOProcessStep$setRationale(rationale, locales = NULL)
```

Arguments:

`rationale` rationale

`locales` list of localized texts. Default is `NULL`

Method `setDateTime()`: Set date time

Usage:

```
ISOProcessStep$setDateTime(dateTime)
```

Arguments:

`dateTime` object of class ISOBaseDateTime or POSIXct

Method `addProcessor()`: Adds processor

Usage:

```
ISOProcessStep$addProcessor(processor)
```

Arguments:

`processor` object of class ISOResponsibleParty

Returns: `TRUE` if added, `FALSE` otherwise

Method `delProcessor()`: Deletes processor

Usage:

```
ISOProcessStep$delProcessor(processor)
```

Arguments:

`processor` object of class ISOResponsibleParty

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addSource()`: Adds source

Usage:

```
ISOProcessStep$addSource(source)
```

Arguments:

`source` object of class ISOSource

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSource()`: Deletes source

Usage:

```
ISOProcessStep$delSource(source)
```

Arguments:

`source` object of class ISOSource

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
ps <- ISOProcessStep$new()
ps$setDescription("description")
ps$setRationale("rationale")
ps$setDateTime( ISOdate(2015, 1, 1, 23, 59, 59))
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone") #and more responsible party properties..
ps$addProcessor(rp)
xml <- ps$encode()
```

ISOProductionSeries *ISOProductionSeries*

Description

ISOProductionSeries

ISOProductionSeries

Format

R6Class object.

Value

Object of R6Class for modelling an ISOProductionSeries

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISOAbstractAggregate -> geometa::ISOSeries -> ISOProductionSeries

Methods

Public methods:

- ISOProductionSeries$new()
- ISOProductionSeries$clone()

Method new()**:** Initializes object

Usage:

```
ISOProductionSeries$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOPropertyType` *ISOPropertyType*

Description

ISOPropertyType

ISOPropertyType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOPropertyType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractCarrierOfCharacteristics` -> `geometa::ISOAbstractPropertyType` -> `ISOPropertyType`

Methods

Public methods:

- `ISOPropertyType$new()`
- `ISOPropertyType$clone()`

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

Usage:

```
ISOPropertyType$new(xml = NULL, defaults = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` default values

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

`ISOQuantitativeAttributeAccuracy`
ISOQuantitativeAttributeAccuracy

Description

ISOQuantitativeAttributeAccuracy

ISOQuantitativeAttributeAccuracy

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOQuantitativeAttributeAccuracy

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISODataQualityAbstractElement -> geometa::ISOAbstractThematicAccuracy -> ISOQuantitativeAttributeAccuracy

Methods

Public methods:

- ISOQuantitativeAttributeAccuracy$clone()

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

Usage:

`ISOQuantitativeAttributeAccuracy$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOQuantitativeAttributeAccuracy$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

`ISOQuantitativeResult` *ISOQuantitativeResult*

Description

ISOQuantitativeResult

ISOQuantitativeResult

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO QuantitativeResult

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractResult` -> `ISOQuantitativeResult`

Public fields

`valueType`	valueType [0..1]- ISORecord
`valueUnit`	valueUnit [1..1]- GMLUnitDefinition
`errorStatistic`	errorStatistic [0..1]
`value`	value [1..*]

Methods

Public methods:

- `ISOQuantitativeResult$new()`
- `ISOQuantitativeResult$setValueType()`
- `ISOQuantitativeResult$setValueUnit()`
- `ISOQuantitativeResult$setErrorStatistic()`
- `ISOQuantitativeResult$addValue()`
- `ISOQuantitativeResult$delValue()`
- `ISOQuantitativeResult$clone()`

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

Usage:

```
ISOQuantitativeResult$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setValueType()`: Set value type

Usage:

```
ISOQuantitativeResult$setValueType(valueType)
```

Arguments:

`valueType` object of class ISORecordType or character

Method `setValueUnit()`: Set value unit

Usage:

```
ISOQuantitativeResult$setValueUnit(valueUnit)
```

Arguments:

`valueUnit` object of class inheriting GMLUnitDefinition

Method `setErrorStatistic()`: Set error statistic

Usage:

```
ISOQuantitativeResult$setErrorStatistic(errorStatistic)
```

Arguments:

`errorStatistic` error statistic

Method `addValue()`: Adds value

Usage:

```
ISOQuantitativeResult$addValue(value)
```

Arguments:

`value` object of class ISORecord or character

Returns: `TRUE` if added, `FALSE` otherwise

Method `delValue()`: Deletes value

Usage:

```
ISOQuantitativeResult$delValue(value)
```

Arguments:

`value` object of class ISORecord or character

Returns: `TRUE` if delete, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOQuantitativeResult$new()
xml <- md$encode()
```

`ISORangeDimension` *ISORangeDimension*

Description

ISORangeDimension

ISORangeDimension

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISORangeDimension

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISORangeDimension`

Public fields

`sequenceIdentifier` sequenceIdentifier

`descriptor` descriptor

Methods

Public methods:

- `ISORangeDimension$new()`
- `ISORangeDimension$setSequenceIdentifier()`
- `ISORangeDimension$setDescriptor()`
- `ISORangeDimension$clone()`

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

Usage:

```
ISORangeDimension$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setSequenceIdentifier():` Set sequence identifier

Usage:

```
ISORangeDimension$setSequenceIdentifier(memberName)
```

Arguments:

`memberName` object of class ISOMemberName

Method `setDescriptor():` Set descriptor

Usage:

```
ISORangeDimension$setDescriptor(descriptor, locales = NULL)
```

Arguments:

`descriptor` descriptor

`locales` list of localized texts. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create dimension
md <- ISORangeDimension$new()
md$setSequenceIdentifier(ISOMemberName$new(aName = "name", attributeType = "type"))
md$setDescriptor("descriptor")
xml <- md$encode()
```

`ISORecord` *ISORecord*

Description

ISORecord

ISORecord

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISORecord

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISORecord`

Public fields

`value` value

Methods

Public methods:

- `ISORecord$new()`
- `ISORecord$clone()`

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

Usage:

```
ISORecord$new(xml = NULL, value)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISORecordType` *ISORecordType*

Description

ISORecordType

ISORecordType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISORecordType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISORecordType`

Public fields

`value` value

Methods

Public methods:

- `ISORecordType$new()`
- `ISORecordType$clone()`

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

Usage:

```
ISORecordType$new(xml = NULL, value)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOReferenceIdentifier` *ISOReferenceIdentifier*

Description

ISOReferenceIdentifier

ISOReferenceIdentifier

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO ReferenceIdentifier

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOIdentifier` -> `ISOReferenceIdentifier`

Public fields

`codeSpace` codeSpace [0..1]: character

`version` version [0..1]: character

Methods

Public methods:

- `ISOReferenceIdentifier$new()`
- `ISOReferenceIdentifier$setCodeSpace()`
- `ISOReferenceIdentifier$setVersion()`
- `ISOReferenceIdentifier$clone()`

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

Usage:

```
ISOReferenceIdentifier$new(xml = NULL, code, codeSpace = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`code` code

`codeSpace` code space

Method `setCodeSpace():` Set code space

Usage:

```
ISOReferenceIdentifier$setCodeSpace(codeSpace)
```

Arguments:

`codeSpace` code space

Method `setVersion():` Set version

Usage:

```
ISOReferenceIdentifier$setVersion(version)
```

Arguments:

`version` version

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOReferenceIdentifier$new(code = "4326", codeSpace = "EPSG")
xml <- md$encode()
```

`ISOReferenceSystem` *ISOReferenceSystem*

Description

ISOReferenceSystem

ISOReferenceSystem

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO ReferenceSystem

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOReferenceSystem`

Public fields

`referenceSystemIdentifier` referenceSystemIdentifier

Methods

Public methods:

- `ISOReferenceSystem$new()`
- `ISOReferenceSystem$setReferenceSystemIdentifier()`
- `ISOReferenceSystem$clone()`

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

Usage:

```
ISOReferenceSystem$new(xml = NULL, prefix, code)
```

Arguments:

`xml` object of class XMLInternalNode-class

`prefix` prefix

`code` code

Method `setReferenceSystemIdentifier()`**:** Set reference system identifier

Usage:

```
ISOReferenceSystem$setReferenceSystemIdentifier(identifier)
```

Arguments:

`identifier` object of class ISOReferenceIdentifier

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

Usage:

`ISOReferenceSystem$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOReferenceSystem$new()
rsId <- ISOReferenceIdentifier$new(code = "4326", codeSpace = "EPSG")
md$setReferenceSystemIdentifier(rsId)
xml <- md$encode()
```

`ISORepresentativeFraction` *ISORepresentativeFraction*

Description

ISORepresentativeFraction

ISORepresentativeFraction

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO RepresentativeFraction

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISORepresentativeFraction`

Public fields

`denominator` denominator

Methods

Public methods:

- `ISORepresentativeFraction$new()`
- `ISORepresentativeFraction$setDenominator()`
- `ISORepresentativeFraction$clone()`

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

Usage:

```
ISORepresentativeFraction$new(xml = NULL, denominator)
```

Arguments:

`xml` object of class XMLInternalNode-class

`denominator` denominator

Method `setDenominator()`**:** Set denominator

Usage:

```
ISORepresentativeFraction$setDenominator(denominator)
```

Arguments:

`denominator` object of class integer

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
fr <- ISORepresentativeFraction$new(denominator = 1L)
xml1 <- fr$encode()
fr$setDenominator(2L)
xml2 <- fr$encode()
```

ISOResolution	*ISOResolution*

Description

ISOResolution

ISOResolution

Format

R6Class object.

Value

Object of R6Class for modelling an ISO Resolution

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOResolution`

Public fields

`equivalentScale` equivalentScale

`distance` distance

Methods

Public methods:

- `ISOResolution$new()`
- `ISOResolution$setEquivalentScale()`
- `ISOResolution$setDistance()`
- `ISOResolution$clone()`

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

Usage:

```
ISOResolution$new(xml = NULL, defaults = list())
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of defaults

Method `setEquivalentScale()`**:** Set equivalent scale

Usage:

```
ISOResolution$setEquivalentScale(equivalentScale)
```

Arguments:

`equivalentScale` object of class ISORepresentativeFraction or numeric

Method `setDistance()`**:** Set distance

Usage:

`ISOResolution$setDistance(distance)`

Arguments:

`distance` object of class ISODistance

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

Usage:

`ISOResolution$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOResolution$new()
md$setDistance(ISODistance$new(value = 1, uom = "m", useUomURI = TRUE))
xml <- md$encode()
```

`ISOResponsibleParty` *ISOResponsibleParty*

Description

ISOResponsibleParty

ISOResponsibleParty

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO ResponsibleParty

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOResponsibleParty`

Public fields

`individualName` individualName

`organisationName` organisationName

`positionName` positionName

`contactInfo` contactInfo

`role` role

Methods

Public methods:

- `ISOResponsibleParty$new()`
- `ISOResponsibleParty$setIndividualName()`
- `ISOResponsibleParty$setOrganisationName()`
- `ISOResponsibleParty$setPositionName()`
- `ISOResponsibleParty$setContactInfo()`
- `ISOResponsibleParty$setRole()`
- `ISOResponsibleParty$clone()`

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

Usage:

```
ISOResponsibleParty$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setIndividualName()`**:** Set individual name

Usage:

```
ISOResponsibleParty$setIndividualName(individualName, locales = NULL)
```

Arguments:

`individualName` individual name

`locales` list of localized texts. Default is `NULL`

Method `setOrganisationName()`**:** Set organisation name

Usage:

```
ISOResponsibleParty$setOrganisationName(organisationName, locales = NULL)
```

Arguments:

`organisationName` organisation name

`locales` list of localized texts. Default is `NULL`

Method `setPositionName()`**:** Set position name

Usage:

`ISOResponsibleParty$setPositionName(positionName, locales = NULL)`

Arguments:

`positionName` position name

`locales` list of localized texts. Default is `NULL`

Method `setContactInfo()`**:** Set contact info

Usage:

`ISOResponsibleParty$setContactInfo(contactInfo)`

Arguments:

`contactInfo` object of class ISOContact

Method `setRole()`**:** Set role

Usage:

`ISOResponsibleParty$setRole(role)`

Arguments:

`role` role object of class ISORole or any character among values returned by `ISORole$values()`

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

Usage:

`ISOResponsibleParty$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create a responsible party element
md <- ISOResponsibleParty$new()
md$setIndividualName("someone")
md$setOrganisationName("somewhere")
md$setPositionName("someposition")
md$setRole("pointOfContact")

#add contact
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
```

```
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact$setOnlineResource(res)
md$setContactInfo(contact)

xml <- md$encode()
```

`ISORestriction` *ISOHierarchyLevel*

Description

ISOHierarchyLevel

ISOHierarchyLevel

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Restriction

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISORestriction`

Methods

Public methods:

- `ISORestriction$new()`
- `ISORestriction$clone()`

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

Usage:

`ISORestriction$new(xml = NULL, value, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISORestriction$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISORestriction$values(labels = TRUE)

#copyright restriction
cr <- ISORestriction$new(value = "copyright")
```

`ISORole` *ISORole*

Description

ISORole

ISORole

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Role

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISORole`

Methods

Public methods:

- `ISORole$new()`
- `ISORole$clone()`

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

Usage:

`ISORole$new(xml = NULL, value = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

`ISORole$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISORole$values(labels = TRUE)

#publisher restriction
role <- ISORole$new(value = "publisher")
```

`ISORoleType` *ISORoleType*

Description

ISORoleType

ISORoleType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO RoleType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISORoleType`

Methods

Public methods:

- `ISORoleType$new()`
- `ISORoleType$clone()`

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

Usage:

```
ISORoleType$new(xml = NULL, value, description = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19110:2005 Methodology for Feature cataloguing

Examples

```
#possible values
values <- ISORoleType$values(labels = TRUE)

#some charset
ordinaryType <- ISORoleType$new(value = "ordinary")
```

`ISOScale` *ISOScale*

Description

ISOScale

ISOScale

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOScale measure

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOMeasure` -> `ISOScale`

Methods

Public methods:

- `ISOScale$new()`
- `ISOScale$clone()`

Method `new()`: Initializes object

Usage:

```
ISOScale$new(xml = NULL, value, uom, useUomURI = FALSE)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`uom` uom symbol of unit of measure used

`useUomURI` use uom URI. Default is `FALSE`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOScope` *ISOScope*

Description

ISOScope

ISOScope

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Scope

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOScope`

Public fields

`level` level

Methods

Public methods:

- `ISOScope$new()`
- `ISOScope$setLevel()`
- `ISOScope$clone()`

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

Usage:

```
ISOScope$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setLevel()`**:** Set level

Usage:

```
ISOScope$setLevel(level)
```

Arguments:

`level` object of class ISOHierarchyLevel or any character among values returned by ISOHierarchyLevel

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOScope$new()
md$setLevel("dataset")
xml <- md$encode()
```

`ISOScopeDescription` *ISOScopeDescription*

Description

ISOScopeDescription

ISOScopeDescription

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO ScopeDescription

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOScopeDescription`

Public fields

`attributes` attributes [1..*]

`features` features [1..*]

`featureInstances` featureInstances [1..*]

`attributeInstances` attributeInstances [1..*]

`dataset` dataset

`other` other

Methods

Public methods:

- `ISOScopeDescription$new()`
- `ISOScopeDescription$addAttribute()`
- `ISOScopeDescription$delAttribute()`
- `ISOScopeDescription$addAttributeInstance()`
- `ISOScopeDescription$delAttributeInstance()`
- `ISOScopeDescription$addFeatureInstance()`
- `ISOScopeDescription$delFeatureInstance()`
- `ISOScopeDescription$setDataset()`
- `ISOScopeDescription$setOther()`
- `ISOScopeDescription$clone()`

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

Usage:

```
ISOScopeDescription$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addAttribute()`**:** Adds attribute

Usage:

```
ISOScopeDescription$addAttribute(attribute)
```

Arguments:

`attribute` attribute

Returns: `TRUE` if added, `FALSE` otherwise

Method `delAttribute()`**:** Deletes attribute

Usage:

```
ISOScopeDescription$delAttribute(attribute)
```

Arguments:

`attribute` attribute

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addAttributeInstance()`**:** Adds attribute instance

Usage:

```
ISOScopeDescription$addAttributeInstance(attributeInstance)
```

Arguments:

`attributeInstance` attribute instance

Returns: `TRUE` if added, `FALSE` otherwise

Method `delAttributeInstance()`**:** Deletes attribute instance

Usage:

```
ISOScopeDescription$delAttributeInstance(attributeInstance)
```

Arguments:

`attributeInstance` attribute instance

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addFeatureInstance()`**:** Adds feature instance

Usage:

```
ISOScopeDescription$addFeatureInstance(featureInstance)
```

Arguments:

`featureInstance` feature instance

Returns: `TRUE` if added, `FALSE` otherwise

Method `delFeatureInstance()`**:** Deletes feature instance

Usage:

```
ISOScopeDescription$delFeatureInstance(featureInstance)
```

Arguments:

`featureInstance` feature instance

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setDataset()`**:** Set dataset

Usage:

```
ISOScopeDescription$setDataset(dataset)
```

Arguments:

`dataset` dataset

Method `setOther()`**:** Set other

Usage:

```
ISOScopeDescription$setOther(other)
```

Arguments:

`other` other

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOScopeDescription$new()
xml <- md$encode()
```

ISOScopedName *ISOScopedName*

Description

ISOScopedName

ISOScopedName

Format

R6Class object.

Value

Object of R6Class for modelling an ISO ScopedName

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::GMLCodeType -> geometa::ISOAbstractGene
-> ISOScopedName

Public fields

`value` value

Methods

Public methods:

- `ISOScopedName$new()`
- `ISOScopedName$clone()`

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

Usage:

```
ISOScopedName$new(xml = NULL, value)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`ISOSecurityConstraints` *ISOSecurityConstraints*

Description

ISOSecurityConstraints

ISOSecurityConstraints

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO SecurityConstraints

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOConstraints` -> `ISOSecurityConstraints`

Public fields

`classification` classification: ISOClassification

`userNote` userNote [0..1]: character

`classificationSystem` classificationSystem [0..1]: character

`handlingDescription` handlingDescription [0..1]: character

Methods

Public methods:

- `ISOSecurityConstraints$new()`
- `ISOSecurityConstraints$setClassification()`
- `ISOSecurityConstraints$setUserNote()`
- `ISOSecurityConstraints$setClassificationSystem()`
- `ISOSecurityConstraints$setHandlingDescription()`
- `ISOSecurityConstraints$clone()`

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

Usage:

```
ISOSecurityConstraints$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setClassification()`**:** Set classification

Usage:

```
ISOSecurityConstraints$setClassification(classification)
```

Arguments:

`classification` object of class ISOClassification or any character among values returned by `ISOClassification$values()`

Method `setUserNote()`**:** Set user note

Usage:

```
ISOSecurityConstraints$setUserNote(userNote, locales = NULL)
```

Arguments:

`userNote` user note

`locales` list of localized texts. Default is `NULL`

Method `setClassificationSystem()`**:** Set classification system

Usage:

```
ISOSecurityConstraints$setClassificationSystem(
  classificationSystem,
  locales = NULL
)
```

Arguments:

`classificationSystem` classification system

`locales` list of localized texts. Default is `NULL`

Method `setHandlingDescription()`**:** Set handling description

Usage:

```
ISOSecurityConstraints$setHandlingDescription(
  handlingDescription,
  locales = NULL
)
```

Arguments:

`handlingDescription` handling description

`locales` list of localized texts. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create object
md <- ISOSecurityConstraints$new()
md$setClassification("secret")
md$setUserNote("ultra secret")
md$setClassificationSystem("no classification in particular")
md$setHandlingDescription("description")

xml <- md$encode()
```

`ISOSensor` *ISOSensor*

Description

ISOSensor

ISOSensor

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOSensor

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractAggregate` -> `geometa::ISOSeries` -> `ISOSensor`

Methods

Public methods:

- `ISOSensor$new()`
- `ISOSensor$clone()`

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

Usage:

```
ISOSensor$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOSeries` *ISOSeries*

Description

ISOSeries

ISOSeries

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOSeries

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractAggregate` -> `ISOSeries`

Methods

Public methods:

- `ISOSeries$new()`
- `ISOSeries$clone()`

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

Usage:

```
ISOSeries$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOServiceIdentification` *ISOServiceIdentification*

Description

ISOServiceIdentification

ISOServiceIdentification

Format

R6Class object.

Value

Object of R6Class for modelling an ISO ServiceIdentification

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::ISOIdentification -> ISOServiceIdentification

Methods

Public methods:

- ISOServiceIdentification$new()
- ISOServiceIdentification$clone()

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

Usage:

`ISOServiceIdentification$new(xml = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

`ISOServiceIdentification$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
md <- ISOServiceIdentification$new()
md$setAbstract("abstract")
md$setPurpose("purpose")

#adding a point of contact
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
```

```
rp$setOrganisationName("somewhere")
rp$setPositionName("someposition")
rp$setRole("pointOfContact")
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact$setOnlineResource(res)
rp$setContactInfo(contact)
md$addPointOfContact(rp)

#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(ISOdate(2015,1,1))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp)
md$setCitation(ct)

#graphic overview
go <- ISOBrowseGraphic$new(
  fileName = "http://wwww.somefile.org/png",
  fileDescription = "Map Overview",
  fileType = "image/png"
)
md$addGraphicOverview(go)

#maintenance information
mi <- ISOMaintenanceInformation$new()
mi$setMaintenanceFrequency("daily")
md$addResourceMaintenance(mi)

#adding legal constraints
lc <- ISOLegalConstraints$new()
lc$addUseLimitation("limitation1")
lc$addUseLimitation("limitation2")
```

```
lc$addUseLimitation("limitation3")
lc$addAccessConstraint("copyright")
lc$addAccessConstraint("license")
lc$addUseConstraint("copyright")
lc$addUseConstraint("license")
md$addResourceConstraints(lc)

xml <- md$encode()
```

`ISOSource` *ISOSource*

Description

ISOSource

ISOSource

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Source

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOSource`

Public fields

`description` description [0..1]: character

`scaleDenominator` scaleDenominator [0..1]: ISORepresentativeFraction

`sourceReferenceSystem` sourceReferenceSystem [0..1]: ISOReferenceSystem

`sourceCitation` sourceCitation [0..1]: ISOCitation

`sourceExtent` sourceExtent [0..*]: ISOExtent

`sourceStep` sourceStep [0..*]: ISOProcessStep

Methods

Public methods:

- `ISOSource$new()`
- `ISOSource$setDescription()`
- `ISOSource$setScaleDenominator()`
- `ISOSource$setReferenceSystem()`

- `ISOSource$setCitation()`
- `ISOSource$addExtent()`
- `ISOSource$delExtent()`
- `ISOSource$addProcessStep()`
- `ISOSource$delProcessStep()`
- `ISOSource$clone()`

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

Usage:

```
ISOSource$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setDescription()`**:** Set description

Usage:

```
ISOSource$setDescription(description, locales = NULL)
```

Arguments:

`description` description

`locales` list of localized texts. Default is `NULL`

Method `setScaleDenominator()`**:** Set scale denominator

Usage:

```
ISOSource$setScaleDenominator(denominator)
```

Arguments:

`denominator` object of class ISORepresentativeFraction

Method `setReferenceSystem()`**:** Set reference system

Usage:

```
ISOSource$setReferenceSystem(referenceSystem)
```

Arguments:

`referenceSystem` object of class ISOReferenceSystem

Method `setCitation()`**:** Set citation

Usage:

```
ISOSource$setCitation(citation)
```

Arguments:

`citation` object of class ISOCitation

Method `addExtent()`**:** Adds extent

Usage:

```
ISOSource$addExtent(extent)
```

Arguments:

`extent` object of class ISOExtent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delExtent()`**:** Deletes extent

Usage:

```
ISOSource$delExtent(extent)
```

Arguments:

`extent` object of class ISOExtent

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addProcessStep()`**:** Adds process step

Usage:

```
ISOSource$addProcessStep(processStep)
```

Arguments:

`processStep` object of class ISOProcessStep

Returns: `TRUE` if added, `FALSE` otherwise

Method `delProcessStep()`**:** Deletes process step

Usage:

```
ISOSource$delProcessStep(processStep)
```

Arguments:

`processStep` object of class ISOProcessStep

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
src <- ISOSource$new()
src$setDescription("description")
src$setScaleDenominator(1L)

rs <- ISOReferenceSystem$new()
rsId <- ISOReferenceIdentifier$new(code = "4326", codeSpace = "EPSG")
rs$setReferenceSystemIdentifier(rsId)
src$setReferenceSystem(rs)

cit <- ISOCitation$new()
cit$setTitle("sometitle") #and more citation properties...
src$setCitation(cit)

extent <- ISOExtent$new()
bbox <- ISOGeographicBoundingBox$new(minx = -180, miny = -90, maxx = 180, maxy = 90)
extent$setGeographicElement(bbox)
src$addExtent(extent)
xml <- src$encode()
```

`ISOSpatialRepresentation` *ISOSpatialRepresentation*

Description

ISOSpatialRepresentation

ISOSpatialRepresentation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO abstract SpatialRepresentation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOSpatialRepresentation`

Methods

Public methods:

- `ISOSpatialRepresentation$new()`
- `ISOSpatialRepresentation$clone()`

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

Usage:

```
ISOSpatialRepresentation$new(xml = NULL, defaults = list())
```

Arguments:

`xml` object of class XMLInternalNode-class

`defaults` list of defaults

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

abstract class

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOSpatialRepresentationType` *ISOSpatialRepresentationType*

Description

ISOSpatialRepresentationType

ISOSpatialRepresentationType

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO SpatialRepresentationType

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOSpatialRepresentationType`

Methods

Public methods:

- `ISOSpatialRepresentationType$new()`
- `ISOSpatialRepresentationType$clone()`

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

Usage:

`ISOSpatialRepresentationType$new(xml = NULL, value = NULL, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOSpatialRepresentationType$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOSpatialRepresentationType$values(labels = TRUE)

#vector example
vectorRep <- ISORestriction$new(value = "vector")
```

`ISOSpatialTemporalExtent` *ISOSpatialTemporalExtent*

Description

ISOSpatialTemporalExtent

ISOSpatialTemporalExtent

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO SpatialTemporalExtent

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOTemporalExtent` -> `ISOSpatialTemporalExtent`

Public fields

`spatialExtent` spatialExtent [1..*]: ISOGeographicExtent

Methods

Public methods:

- `ISOSpatialTemporalExtent$new()`
- `ISOSpatialTemporalExtent$addSpatialExtent()`
- `ISOSpatialTemporalExtent$delSpatialExtent()`
- `ISOSpatialTemporalExtent$clone()`

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

Usage:

```
ISOSpatialTemporalExtent$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `addSpatialExtent()`**:** Adds spatial extent

Usage:

```
ISOSpatialTemporalExtent$addSpatialExtent(spatialExtent)
```

Arguments:

`spatialExtent` object of class ISOGeographicExtent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delSpatialExtent()`**:** Deletes spatial extent

Usage:

```
ISOSpatialTemporalExtent$delSpatialExtent(spatialExtent)
```

Arguments:

`spatialExtent` object of class ISOGeographicExtent

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOSpatialTemporalExtent$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#create object
md <- ISOSpatialTemporalExtent$new()
start <- ISOdate(2000, 1, 12, 12, 59, 45)
end <- ISOdate(2010, 8, 22, 13, 12, 43)
tp <- GMLTimePeriod$new(beginPosition = start, endPosition = end)
md$setTimePeriod(tp)
spatialExtent <- ISOGeographicBoundingBox$new(minx = -180, miny = -90, maxx = 180, maxy = 90)
md$addSpatialExtent(spatialExtent)

xml <- md$encode()
```

`ISOSRVServiceIdentification` *ISOSRVServiceIdentification*

Description

ISOSRVServiceIdentification

ISOSRVServiceIdentification

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO ServiceIdentification

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOIdentification` -> `geometa::ISOServiceIdentification` -> `ISOSRVServiceIdentification`

Public fields

`serviceType` serviceType [1..1]: ISOGenericName

`serviceTypeVersion` serviceTypeVersion [0..*]: character

`accessProperties` accessProperties [0..1]: ISOStandardOrderProcess

`restrictions` restrictions [0..1]: ISOConstraints

`keywords` keywords [0..*]: ISOKeywords

`extent` extent [0..*]: ISOExtent

`coupledResource` coupledResource [0..*]: ISOCoupledResource

`couplingType` couplingType [1..1]: ISOCouplingType

`containsOperations` containsOperations [1..*]: ISOOperationMetadata

`operatesOn` operatesOn [0..*]: ISODataIdentification

Methods

Public methods:

- `ISOSRVServiceIdentification$new()`
- `ISOSRVServiceIdentification$setServiceType()`
- `ISOSRVServiceIdentification$addServiceTypeVersion()`
- `ISOSRVServiceIdentification$delServiceTypeVersion()`
- `ISOSRVServiceIdentification$setAccessProperties()`
- `ISOSRVServiceIdentification$setRestrictions()`
- `ISOSRVServiceIdentification$addKeywords()`
- `ISOSRVServiceIdentification$delKeywords()`
- `ISOSRVServiceIdentification$addExtent()`
- `ISOSRVServiceIdentification$delExtent()`
- `ISOSRVServiceIdentification$addCoupledResource()`
- `ISOSRVServiceIdentification$delCoupledResource()`
- `ISOSRVServiceIdentification$setCouplingType()`
- `ISOSRVServiceIdentification$addOperationMetadata()`
- `ISOSRVServiceIdentification$delOperationMetadata()`
- `ISOSRVServiceIdentification$addOperatesOn()`
- `ISOSRVServiceIdentification$delOperatesOn()`
- `ISOSRVServiceIdentification$clone()`

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

Usage:

```
ISOSRVServiceIdentification$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setServiceType()`**:** Set service type

Usage:
`ISOSRVServiceIdentification$setServiceType(serviceType)`

Arguments:
`serviceType` object of class ISOLocalName, ISOScopedName or character

Method `addServiceTypeVersion()`**:** Adds service type version

Usage:
`ISOSRVServiceIdentification$addServiceTypeVersion(version)`

Arguments:
`version` version

Returns: `TRUE` if added, `FALSE` otherwise

Method `delServiceTypeVersion()`**:** Deletes service type version

Usage:
`ISOSRVServiceIdentification$delServiceTypeVersion(version)`

Arguments:
`version` version

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setAccessProperties()`**:** Set access properties

Usage:
`ISOSRVServiceIdentification$setAccessProperties(accessProperties)`

Arguments:
`accessProperties` object of class ISOStandardOrderProcess

Method `setRestrictions()`**:** Set restrictions

Usage:
`ISOSRVServiceIdentification$setRestrictions(restrictions)`

Arguments:
`restrictions` object of class ISOConstraints

Method `addKeywords()`**:** Adds keywords

Usage:
`ISOSRVServiceIdentification$addKeywords(keywords)`

Arguments:
`keywords` object of class ISOKeywords

Returns: `TRUE` if added, `FALSE` otherwise

Method `delKeywords()`**:** Deletes keywords

Usage:
`ISOSRVServiceIdentification$delKeywords(keywords)`

Arguments:

`keywords` object of class ISOKeywords

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addExtent()`**:** Adds extent

Usage:

```
ISOSRVServiceIdentification$addExtent(extent)
```

Arguments:

`extent` object of class ISOExtent

Returns: `TRUE` if added, `FALSE` otherwise

Method `delExtent()`**:** Deletes extent

Usage:

```
ISOSRVServiceIdentification$delExtent(extent)
```

Arguments:

`extent` object of class ISOExtent

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addCoupledResource()`**:** Adds coupled resource

Usage:

```
ISOSRVServiceIdentification$addCoupledResource(resource)
```

Arguments:

`resource` object of class ISOCoupledResource

Returns: `TRUE` if added, `FALSE` otherwise

Method `delCoupledResource()`**:** Deletes coupled resource

Usage:

```
ISOSRVServiceIdentification$delCoupledResource(resource)
```

Arguments:

`resource` object of class ISOCoupledResource

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `setCouplingType()`**:** Set coupling type

Usage:

```
ISOSRVServiceIdentification$setCouplingType(couplingType)
```

Arguments:

`couplingType` object of class ISOCouplingType or any character among values returned by `ISOCouplingType$values()`

Method `addOperationMetadata()`**:** Adds operation metadata

Usage:

```
ISOSRVServiceIdentification$addOperationMetadata(operationMetadata)
```

Arguments:

`operationMetadata` object of class ISOOperationMetadata

Returns: `TRUE` if added, `FALSE` otherwise

Method `delOperationMetadata():` Deletes operation metadata

Usage:

```
ISOSRVServiceIdentification$delOperationMetadata(operationMetadata)
```

Arguments:

`operationMetadata` object of class ISOOperationMetadata

Returns: `TRUE` if deleted, `FALSE` otherwise

Method `addOperatesOn():` Adds operates on

Usage:

```
ISOSRVServiceIdentification$addOperatesOn(dataIdentification)
```

Arguments:

`dataIdentification` object of class ISODataIdentification

Returns: `TRUE` if added, `FALSE` otherwise

Method `delOperatesOn():` Deletes operates on

Usage:

```
ISOSRVServiceIdentification$delOperatesOn(dataIdentification)
```

Arguments:

`dataIdentification` object of class ISODataIdentification

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19119:2005 - Geographic information – Services

Examples

```
#encoding
md <- ISOSRVServiceIdentification$new()
md$setAbstract("abstract")
md$setPurpose("purpose")

#adding a point of contact
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
rp$setOrganisationName("somewhere")
rp$setPositionName("someposition")
rp$setRole("pointOfContact")
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenumber")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somename")
contact$setOnlineResource(res)
rp$setContactInfo(contact)
md$addPointOfContact(rp)

#citation
ct <- ISOCitation$new()
ct$setTitle("sometitle")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
ct$addDate(d)
ct$setEdition("1.0")
ct$setEditionDate(ISOdate(2015,1,1))
ct$addIdentifier(ISOMetaIdentifier$new(code = "identifier"))
ct$addPresentationForm("mapDigital")
ct$addCitedResponsibleParty(rp)
md$setCitation(ct)

#graphic overview
go <- ISOBrowseGraphic$new(
  fileName = "http://wwww.somefile.org/png",
  fileDescription = "Map Overview",
  fileType = "image/png"
)
md$addGraphicOverview(go)
```

```
#maintenance information
mi <- ISOMaintenanceInformation$new()
mi$setMaintenanceFrequency("daily")
md$addResourceMaintenance(mi)

#adding legal constraints
lc <- ISOLegalConstraints$new()
lc$addUseLimitation("limitation1")
lc$addUseLimitation("limitation2")
lc$addUseLimitation("limitation3")
lc$addAccessConstraint("copyright")
lc$addAccessConstraint("license")
lc$addUseConstraint("copyright")
lc$addUseConstraint("license")
md$addResourceConstraints(lc)

#specific elements to service identification
md$setServiceType("Fishery data harmonization process")
md$addServiceTypeVersion("1.0")
orderProcess <- ISOStandardOrderProcess$new()
orderProcess$setFees("fees")
orderProcess$setPlannedAvailableDateTime(ISOdate(2017,7,5,12,0,0))
orderProcess$setOrderingInstructions("instructions")
orderProcess$setTurnaround("turnaround")
md$setAccessProperties(orderProcess)
md$setRestrictions(lc)

kwds <- ISOKeywords$new()
kwds$addKeyword("keyword1")
kwds$addKeyword("keyword2")
kwds$setKeywordType("theme")
th <- ISOCitation$new()
th$setTitle("General")
th$addDate(d)
kwds$setThesaurusName(th)
md$addKeywords(kwds)

#adding extent
extent <- ISOExtent$new()
bbox <- ISOGeographicBoundingBox$new(minx = -180, miny = -90, maxx = 180, maxy = 90)
extent$addGeographicElement(bbox)
md$addExtent(extent)

#coupling type
#(here "tight" associated with a particular dataset "my-dataset-identifier")
#see ISOCouplingType$values(labels = T) for other values
md$setCouplingType("tight")
coupledDataset1 <- ISOCoupledResource$new()
coupledDataset1$setOperationName("Rscript")
coupledDataset1$setIdentifier("my-dataset-identifier")
coupledDataset2 <- ISOCoupledResource$new()
coupledDataset2$setOperationName("WPS:Execute")
```

```
coupledDataset2$setIdentifier("my-dataset-identifier")
md$addCoupledResource(coupledDataset1)
md$addCoupledResource(coupledDataset2)

#add operation metadata 1 (Rscript)
scriptOp <- ISOOperationMetadata$new()
scriptOp$setOperationName("Rscript")
scriptOp$addDCP("WebServices")
scriptOp$setOperationDescription("WPS Execute")
scriptOp$setInvocationName("identifier")
for(i in 1:3){
  param <- ISOParameter$new()
  param$setName(sprintf("name%s",i), "xs:string")
  param$setDirection("in")
  param$setDescription(sprintf("description%s",i))
  param$setOptionality(FALSE)
  param$setRepeatability(FALSE)
  param$setValueType("xs:string")
  scriptOp$addParameter(param)
}
outParam <-ISOParameter$new()
outParam$setName("outputname", "xs:string")
outParam$setDirection("out")
outParam$setDescription("outputdescription")
outParam$setOptionality(FALSE)
outParam$setRepeatability(FALSE)
outParam$setValueType("xs:string")
scriptOp$addParameter(outParam)
or <- ISOOnlineResource$new()
or$setLinkage("http://somelink/myrscript.R")
or$setName("R script name")
or$setDescription("R script description")
or$setProtocol("protocol")
scriptOp$addConnectPoint(or)
md$addOperationMetadata(scriptOp)
#add operation metadata 1 (WPS)
wpsOp <- ISOOperationMetadata$new()
wpsOp$setOperationName("WPS:Execute")
wpsOp$addDCP("WebServices")
wpsOp$setOperationDescription("WPS Execute")
invocationName <- "mywpsidentifier"
wpsOp$setInvocationName(invocationName)
for(i in 1:3){
  param <- ISOParameter$new()
  param$setName(sprintf("name%s",i), "xs:string")
  param$setDirection("in")
  param$setDescription(sprintf("description%s",i))
  param$setOptionality(FALSE)
  param$setRepeatability(FALSE)
  param$setValueType("xs:string")
  wpsOp$addParameter(param)
}
outParam <-ISOParameter$new()
```

```
outParam$setName("outputname", "xs:string")
outParam$setDirection("out")
outParam$setDescription("outputdescription")
outParam$setOptionality(FALSE)
outParam$setRepeatability(FALSE)
outParam$setValueType("xs:string")
wpsOp$addParameter(outParam)
or1 <- ISOOnlineResource$new()
or1$setLinkage(
  sprintf("http://somelink/wps?request=Execute&version=1.0.0&Identifier=%s",
  invocationName)
)
or1$setName("WPS process name")
or1$setDescription("WPS process description")
or1$setProtocol("protocol")
wpsOp$addConnectPoint(or1)
or2 <- ISOOnlineResource$new()
or2$setLinkage("http://somelink/myrscript.R")
or2$setName("Source R script name")
or2$setDescription("Source R script description")
or2$setProtocol("protocol")
wpsOp$addConnectPoint(or2)
md$addOperationMetadata(wpsOp)
xml <- md$encode()
```

`ISOStandardOrderProcess` *ISOStandardOrderProcess*

Description

ISOStandardOrderProcess

ISOStandardOrderProcess

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO StandardOrderProcess

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOStandardOrderProcess`

Public fields

`fees` fees [0..1]: character

`plannedAvailableDateTime` plannedAvailableDateTime [0..1]: 'POSIXct/POSIXlt'

`orderingInstructions` orderingInstructions [0..1]: character

`turnaround` turnaround [0..1]: character

Methods

Public methods:

- `ISOStandardOrderProcess$new()`
- `ISOStandardOrderProcess$setFees()`
- `ISOStandardOrderProcess$setPlannedAvailableDateTime()`
- `ISOStandardOrderProcess$setOrderingInstructions()`
- `ISOStandardOrderProcess$setTurnaround()`
- `ISOStandardOrderProcess$clone()`

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

Usage:

```
ISOStandardOrderProcess$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setFees()`**:** Set fees

Usage:

```
ISOStandardOrderProcess$setFees(fees, locales = NULL)
```

Arguments:

`fees` fees

`locales` list of localized texts. Default is `NULL`

Method `setPlannedAvailableDateTime()`**:** Set planned available date time

Usage:

```
ISOStandardOrderProcess$setPlannedAvailableDateTime(dateTime)
```

Arguments:

`dateTime` object of class POSIXct

Method `setOrderingInstructions()`**:** Set ordering instructions

Usage:

```
ISOStandardOrderProcess$setOrderingInstructions(instructions, locales = NULL)
```

Arguments:

`instructions` instructions

`locales` list of localized texts. Default is `NULL`

Method `setTurnaround()`**:** Set turnaround

Usage:

`ISOStandardOrderProcess$setTurnaround(turnaround, locales = NULL)`

Arguments:

`turnaround` turnaround

`locales` list of localized texts. Default is `NULL`

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

Usage:

`ISOStandardOrderProcess$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOStandardOrderProcess$new()
md$setFees("fees")
md$setPlannedAvailableDateTime(ISOdate(2017,7,5,12,0,0))
md$setOrderingInstructions("instructions")
md$setTurnaround("turnaround")
xml <- md$encode()
```

`ISOStatus` *ISOStatus*

Description

ISOStatus

ISOStatus

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO progress status

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOStatus`

Methods

Public methods:

- `ISOStatus$new()`
- `ISOStatus$clone()`

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

Usage:

`ISOStatus$new(xml = NULL, value, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOStatus$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOStatus$values(labels = TRUE)

#pending status
pending <- ISOStatus$new(value = "pending")
```

`ISOStereoMate` *ISOStereoMate*

Description

ISOStereoMate

ISOStereoMate

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOStereoMate

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOAbstractAggregate` -> `ISOStereoMate`

Methods

Public methods:

- `ISOStereoMate$new()`
- `ISOStereoMate$clone()`

Method `new()`**:** Initialize object

Usage:

```
ISOStereoMate$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOTelephone` *ISOTelephone*

Description

ISOTelephone

ISOTelephone

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Telephone

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOTelephone`

Public fields

`voice` voice

`facsimile` facsimile

Methods

Public methods:

- `ISOTelephone$new()`
- `ISOTelephone$setVoice()`
- `ISOTelephone$setFacsimile()`
- `ISOTelephone$clone()`

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

Usage:

```
ISOTelephone$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setVoice()`**:** Set voice

Usage:

```
ISOTelephone$setVoice(voice, locales = NULL)
```

Arguments:

`voice` voice

`locales` list of localized voices. Default is `NULL`

Method `setFacsimile():` Set facsimile

Usage:

```
ISOTelephone$setFacsimile(facsimile, locales = NULL)
```

Arguments:

`facsimile` facsimile

`locales` list of localized facsimiles. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOTelephone$new()
md$setVoice("myphonenumber")
md$setFacsimile("myfacsimile")
xml <- md$encode()
```

`ISOTemporalConsistency` *ISOTemporalConsistency*

Description

ISOTemporalConsistency

ISOTemporalConsistency

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOTemporalConsistency

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `geometa::ISOAbstractTemporalAccuracy` -> `ISOTemporalConsistency`

Methods

Public methods:

- `ISOTemporalConsistency$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOTemporalConsistency$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

`ISOTemporalExtent` *ISOTemporalExtent*

Description

ISOTemporalExtent

ISOTemporalExtent

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO TemporalExtent

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOTemporalExtent`

Public fields

`extent` extent

Methods

Public methods:

- `ISOTemporalExtent$new()`
- `ISOTemporalExtent$setTimeInstant()`
- `ISOTemporalExtent$setTimePeriod()`
- `ISOTemporalExtent$clone()`

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

Usage:

```
ISOTemporalExtent$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setTimeInstant()`**:** Set time instant

Usage:

```
ISOTemporalExtent$setTimeInstant(timeInstant)
```

Arguments:

`timeInstant` object of class GMLTimeInstant

Method `setTimePeriod()`**:** Set time period

Usage:

`ISOTemporalExtent$setTimePeriod(timePeriod)`

Arguments:

`timePeriod` object of class GMLTimePeriod

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

Usage:

`ISOTemporalExtent$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
te <- ISOTemporalExtent$new()
start <- ISOdate(2000, 1, 12, 12, 59, 45)
end <- ISOdate(2010, 8, 22, 13, 12, 43)
tp <- GMLTimePeriod$new(beginPosition = start, endPosition = end)
te$setTimePeriod(tp)
```

`ISOTemporalValidity` *ISOTemporalValidity*

Description

ISOTemporalValidity

ISOTemporalValidity

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOTemporalValidity

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `geometa::ISOAbstractTemporalAccuracy` -> `ISOTemporalValidity`

Methods

Public methods:

- `ISOTemporalValidity$clone()`

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

Usage:

`ISOTemporalValidity$clone(deep = FALSE)`

Arguments:

deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOTemporalValidity$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

`ISOThematicClassificationCorrectness`
ISOThematicClassificationCorrectness

Description

ISOThematicClassificationCorrectness

ISOThematicClassificationCorrectness

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOThematicClassificationCorrectness

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `geometa::ISOAbstractTemporalAccuracy` -> `ISOThematicClassificationCorrectness`

Methods

Public methods:

- `ISOThematicClassificationCorrectness$clone()`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOThematicClassificationCorrectness$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

`ISOTopicCategory` *ISOTopicCategory*

Description

ISOTopicCategory

ISOTopicCategory

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO TopicCategory

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOTopicCategory`

Methods

Public methods:

- `ISOTopicCategory$new()`
- `ISOTopicCategory$clone()`

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

Usage:

`ISOTopicCategory$new(xml = NULL, value, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOTopicCategory$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOTopicCategory$values(labels = TRUE)

#biota topic
biota <- ISOTopicCategory$new(value = "biota")
```

`ISOTopologicalConsistency` *ISOTopologicalConsistency*

Description

ISOTopologicalConsistency

ISOTopologicalConsistency

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOTopologicalConsistency

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISODataQualityAbstractElement` -> `geometa::ISOAbstractLogicalConsistency` -> `ISOTopologicalConsistency`

Methods

Public methods:

- `ISOTopologicalConsistency$clone()`

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

Usage:

`ISOTopologicalConsistency$clone(deep = FALSE)`

Arguments:

deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#encoding
dq <- ISOTopologicalConsistency$new()
dq$addNameOfMeasure("measure")
metaId <- ISOMetaIdentifier$new(code = "measure-id")
dq$setMeasureIdentification(metaId)
dq$setMeasureDescription("description")
dq$setEvaluationMethodDescription("method description")
dq$setEvaluationMethodType("indirect")
dq$setDateTime(ISOdate(2015,1,1,12,10,49))
spec <- ISOCitation$new()
spec$setTitle("specification title")
spec$addAlternateTitle("specification alternate title")
d <- ISODate$new()
d$setDate(ISOdate(2015, 1, 1, 1))
d$setDateType("publication")
spec$addDate(d)
```

```
dq$setEvaluationProcedure(spec)
result <- ISOConformanceResult$new()
result$setSpecification(spec)
result$setExplanation("some explanation about the conformance")
result$setPass(TRUE)
dq$addResult(result)
xml <- dq$encode()
```

`ISOTopologyLevel` *ISOTopologyLevel*

Description

ISOTopologyLevel

ISOTopologyLevel

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO TopologyLevel

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOCodeListValue` -> `ISOTopologyLevel`

Methods

Public methods:

- `ISOTopologyLevel$new()`
- `ISOTopologyLevel$clone()`

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

Usage:

`ISOTopologyLevel$new(xml = NULL, value, description = NULL)`

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

`description` description

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

Usage:

`ISOTopologyLevel$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
#possible values
values <- ISOTopologyLevel$values(labels = TRUE)

#geomOnly
geomOnly <- ISOTopologyLevel$new(value = "geometryOnly")
```

`ISOTypeName` *ISOTypeName*

Description

ISOTypeName

ISOTypeName

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOTypeName

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOTypeName`

Public fields

`aName` aName: character

Methods

Public methods:

- `ISOTypeName$new()`
- `ISOTypeName$setName()`
- `ISOTypeName$clone()`

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

Usage:

```
ISOTypeName$new(xml = NULL, aName = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`aName` name

Method `setName()`**:** Set name

Usage:

```
ISOTypeName$setName(aName, locales = NULL)
```

Arguments:

`aName` name

`locales` list of localized names. Default is `NULL`

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

Examples

```
typeName <- ISOTypeName$new(aName = "name")
xml <- typeName$encode()
```

`ISOUnlimitedInteger` *ISOUnlimitedInteger*

Description

ISOUnlimitedInteger

ISOUnlimitedInteger

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO UnlimitedInteger

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOUnlimitedInteger`

Public fields

`value` value

`attrs` attrs

Methods

Public methods:

- `ISOUnlimitedInteger$new()`
- `ISOUnlimitedInteger$clone()`

Method `new()`**:** Initialize object

Usage:

```
ISOUnlimitedInteger$new(xml = NULL, value)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

ISOURL *ISOURL*

Description

ISOURL

ISOURL

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISOURL

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOURL`

Public fields

`value` value

Methods

Public methods:

- `ISOURL$new()`
- `ISOURL$setUrl()`
- `ISOURL$clone()`

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

Usage:

```
ISOURL$new(xml = NULL, value = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

`value` value

Method `setUrl()`**:** Set URL

Usage:

```
ISOURL$setUrl(url)
```

Arguments:

`url` url

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOUsage` *ISOUsage*

Description

ISOUsage

ISOUsage

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO Usage

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOUsage`

Public fields

`specificUsage` specificUsage

`usageDateTime` usageDateTime

`userDeterminedLimitations` userDeterminedLimitations

`userContactInfo` userContactInfo

Methods

Public methods:

- `ISOUsage$new()`
- `ISOUsage$setSpecificUsage()`
- `ISOUsage$setUsageDateTime()`
- `ISOUsage$setUserDeterminedLimitations()`
- `ISOUsage$addUserContact()`
- `ISOUsage$delUserContact()`
- `ISOUsage$clone()`

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

Usage:

```
ISOUsage$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setSpecificUsage()`**:** Set specificUsage

Usage:

```
ISOUsage$setSpecificUsage(specificUsage, locales = NULL)
```

Arguments:

`specificUsage` specific usage

`locales` list of localized texts. Default is `NULL`

Method `setUsageDateTime()`**:** Set usage date time

Usage:

```
ISOUsage$setUsageDateTime(usageDateTime)
```

Arguments:

`usageDateTime` object of class POSIXct

Method `setUserDeterminedLimitations()`**:** Set user determined limitations

Usage:

```
ISOUsage$setUserDeterminedLimitations(
  userDeterminedLimitations,
  locales = NULL
)
```

Arguments:

`userDeterminedLimitations` user determined limitations

`locales` list of localized texts. Default is `NULL`

Method `addUserContact()`**:** Adds user contact

Usage:

```
ISOUsage$addUserContact(contact)
```

Arguments:

`contact` object of class ISOResponsibleParty

Returns: `TRUE` if added, `FALSE` otherwise

Method `delUserContact()`**:** Deletes user contact

Usage:

```
ISOUsage$delUserContact(contact)
```

Arguments:

`contact` object of class ISOResponsibleParty

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

`ISOVectorSpatialRepresentation` *ISOVectorSpatialRepresentation*

Description

ISOVectorSpatialRepresentation

ISOVectorSpatialRepresentation

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO VectorSpatialRepresentation

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::ISOSpatialRepresentation` -> `ISOVectorSpatialRepresentation`

Public fields

`topologyLevel` topologyLevel [0..1]: ISOTopologyLevel

`geometricObjects` geometricObjects [0..*]: ISOGeometricObjects

Methods

Public methods:

- `ISOVectorSpatialRepresentation$new()`
- `ISOVectorSpatialRepresentation$setTopologyLevel()`
- `ISOVectorSpatialRepresentation$addGeometricObjects()`
- `ISOVectorSpatialRepresentation$setGeometricObjects()`
- `ISOVectorSpatialRepresentation$delGeometricObjects()`
- `ISOVectorSpatialRepresentation$clone()`

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

Usage:

```
ISOVectorSpatialRepresentation$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setTopologyLevel()`**:** Set topology level

Usage:

```
ISOVectorSpatialRepresentation$setTopologyLevel(topologyLevel)
```

Arguments:

`topologyLevel` object of class ISOTopologyLevel or character among values returned by `ISOTopologyLevel$values(`

Method `addGeometricObjects()`**:** Adds geometric objects

Usage:

```
ISOVectorSpatialRepresentation$addGeometricObjects(geometricObjects)
```

Arguments:

`geometricObjects` geometric objects, object of ISOGeometricObjects

Returns: `TRUE` if added, `FALSE` otherwise

Method `setGeometricObjects()`**:** Set geometric objects

Usage:

```
ISOVectorSpatialRepresentation$setGeometricObjects(geometricObjects)
```

Arguments:

`geometricObjects` geometric objects, object of ISOGeometricObjects

Returns: `TRUE` if set, `FALSE` otherwise

Method `delGeometricObjects()`**:** Deletes geometric objects

Usage:

`ISOVectorSpatialRepresentation$delGeometricObjects(geometricObjects)`

Arguments:

`geometricObjects` geometric objects, object of ISOGeometricObjects

Returns: `TRUE` if deleted, `FALSE` otherwise

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

Usage:

`ISOVectorSpatialRepresentation$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
md <- ISOVectorSpatialRepresentation$new()
md$setTopologyLevel("geometryOnly")
geomObject1 <- ISOGeometricObjects$new()
geomObject1$setGeometricObjectType("surface")
geomObject1$setGeometricObjectCount(5L)
md$addGeometricObjects(geomObject1)
xml <- md$encode()
```

`ISOVerticalExtent` *ISOVerticalExtent*

Description

ISOVerticalExtent

ISOVerticalExtent

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an ISO VerticalExtent

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `ISOVerticalExtent`

Public fields

`minimumValue` minimumValue [1..1]: numeric

`maximumValue` maximumValue [1..1]: numeric

`unitOfMeasure` unitOfMeasure [1..1]: character

`verticalCRS` verticalCRS [1..1]: GMLVerticalCRS

Methods

Public methods:

- `ISOVerticalExtent$new()`
- `ISOVerticalExtent$setMinimumValue()`
- `ISOVerticalExtent$setMaximumValue()`
- `ISOVerticalExtent$setUnitOfMeasure()`
- `ISOVerticalExtent$setVerticalCRS()`
- `ISOVerticalExtent$clone()`

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

Usage:

```
ISOVerticalExtent$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class

Method `setMinimumValue()`**:** Set minimum value

Usage:

```
ISOVerticalExtent$setMinimumValue(minimumValue)
```

Arguments:

`minimumValue` minimum value

Method `setMaximumValue()`**:** Set maximum value

Usage:

```
ISOVerticalExtent$setMaximumValue(maximumValue)
```

Arguments:

`maximumValue` maximum value

Method `setUnitOfMeasure()`**:** Set unit of measure

Usage:

```
ISOVerticalExtent$setUnitOfMeasure(uom)
```

Arguments:

`uom` uom

Method `setVerticalCRS()`**:** Set vertical CRS

Usage:

`ISOVerticalExtent$setVerticalCRS(verticalCRS)`

Arguments:

`verticalCRS` verticalCRS

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

Usage:

`ISOVerticalExtent$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO 19115:2003 - Geographic information – Metadata

Examples

```
ve <- ISOVerticalExtent$new()
ve$setMinimumValue(0)
ve$setMaximumValue(19)
xml <- ve$encode()
```

`pivot_converter` *pivot_converter*

Description

pivot_converter

pivot_converter

Format

`R6Class` object.

Value

Object of `R6Class` for modelling a mapping format converter

Public fields

`from` from

`to` to

Methods

Public methods:

- `pivot_converter$new()`
- `pivot_converter$clone()`

Method `new()`**:** Initializes pivot converter

Usage:

```
pivot_converter$new(from, to)
```

Arguments:

`from` from

`to` to

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

`pivot_format` *pivot_format*

Description

pivot_format

pivot_format

Format

`R6Class` object.

Value

Object of `R6Class` for modelling a mapping format

Public fields

`id` id

`pkg` pkg

`reader` reader

`checker` checker

`constructor` constructor

Methods

Public methods:

- `pivot_format$new()`
- `pivot_format$clone()`

Method `new()`: Initializes pivot format. Method is used to instantiate a pivot_format, given a unique `id`, the name of package used (for information only). A format is then defined by string expressions (using `sprintf` formatting) to read metadata properties (`reader`), one for checking existence of properties (`checker`), and an expression to create metadata objects (`constructor`). In case the `constructor` is NULL, then no conversion to this metadata format will be possible.

Usage:

```
pivot_format$new(id, pkg, reader = NULL, checker = NULL, constructor = NULL)
```

Arguments:

`id` id

`pkg` pkg

`reader` reader

`checker` checker

`constructor` constructor

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```
#example on how geometa format is defined as pivot format
pivot_format$new(
  id = "geometa", pkg = "geometa",
  reader = "%s[[%s]]", checker = "!is.null(%s[[%s]])",
  constructor = "ISOMetadata$new"
)
```

readISO19139 *readISO19139*

Description

readISO19139 is a function to read a ISO 19139 from a file or url into an object in the **geometa** model.

Usage

```
readISO19139(file, url, raw)
```

Arguments

file	a valid file path, as object of class character
url	a valid URL, as object of class character
raw	indicates if the function should return the raw XML. By default this is set to FALSE and the function will try to map the xml data to the **geometa** data model.

Value

a **geometa** object inheriting ISOAbstractObject

Author(s)

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

Examples

```
  mdfile <- system.file("extdata/examples", "metadata.xml", package = "geometa")
  md <- readISO19139(mdfile)
```

registerISOCodelist *registerISOCodelist*

Description

registerISOCodelist allows to register a new codelist registered in **geometa**

Usage

```
registerISOCodelist(refFile, id, force)
```

Arguments

`refFile`	ISO XML file handling the ISO codelist
`id`	identifier of the ISO codelist
`force`	logical parameter indicating if registration has be to be forced in case the identified codelist is already registered

Author(s)

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

Examples

```
   registerISOCodelist(
    refFile = "http://www.isotc211.org/2005/resources/Codelist/ML_gmxCodelists.xml",
    id = "LanguageCode",
    force = TRUE
  )
```

`registerISOMetadataNamespace`

registerISOMetadataNamespace

Description

`registerISOMetadataNamespace` allows to register a new namespace in **geometa**

Usage

```
registerISOMetadataNamespace(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

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

registerISOMetadataSchema *registerISOMetadataSchema*

Description

registerISOMetadataSchema allows to register a new schema in **geometa**

Usage

```
registerISOMetadataSchema(xsdFile)
```

Arguments

xsdFile	the schema XSD file

Author(s)

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

Examples

```
registerISOMetadataSchema(xsdFile = "http://www.isotc211.org/2005/gmd/gmd.xsd")
```

registerMappingFormat *registerMappingFormat*

Description

registerMappingFormat allows to register a new mapping format in **geometa**

Usage

```
registerMappingFormat(mapping_format)
```

Arguments

mapping_format	object of class pivot_format

Author(s)

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

`registerMappings` *registerMappings*

Description

`MappingFile` allows to register in **geometa** a `data.frame` containing mappings rules to convert from/to other metadata formats (currently EML/emld objects and NetCDF-CF/ncdf4 objects)

Usage

```
registerMappings(x)
```

Arguments

`x`	a `data.frame` containing the metadata mapping rules

`setGeometaOption` *setGeometaOption*

Description

`setGeometaOption` allows to set an option from **geometa**

Usage

```
setGeometaOption(option, value)
```

Arguments

`option`	the name of the option
`value`	the value to set for the option

Author(s)

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

Examples

```
setGeometaOption("schemaBaseUrl", "http://somealternativeurl")
```

`setIANAMimeTypes` *setIANAMimeTypes*

Description

setIANAMimeTypes

Usage

```
setIANAMimeTypes()
```

`setISOCodelists` *setISOCodelists*

Description

setISOCodelists

Usage

```
setISOCodelists()
```

`setISOMetadataNamespaces` *setMetadataNamespaces*

Description

setMetadataNamespaces

Usage

```
setISOMetadataNamespaces()
```

`setISOMetadataSchemas` *setISOMetadataSchemas*

Description

setISOMetadataSchemas

Usage

```
setISOMetadataSchemas()
```

`setMappingFormats` *setMappingFormats*

Description

setMappingFormats

Usage

```
setMappingFormats()
```

`SWEAbstractDataComponent` *SWEAbstractDataComponent*

Description

SWEAbstractDataComponent

SWEAbstractDataComponent

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE Abstract data component

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractSWEIdentifiable` -> `SWEAbstractDataComponent`

Public fields

`name` name

Methods

Public methods:

- `SWEAbstractDataComponent$new()`
- `SWEAbstractDataComponent$addName()`
- `SWEAbstractDataComponent$delName()`
- `SWEAbstractDataComponent$clone()`

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

Usage:

```
SWEAbstractDataComponent$new(
  xml = NULL,
  element = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

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

`element` element

`updatable` updatable

`optional` optional

`definition` definition

Method `addName()`**:** Adds name

Usage:

```
SWEAbstractDataComponent$addName(name, codeSpace = NULL)
```

Arguments:

`name` name

`codeSpace` codespace

Method `delName()`**:** Deletes name

Usage:

```
SWEAbstractDataComponent$delName(name, codeSpace = NULL)
```

Arguments:

`name` name

`codeSpace` codespace

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

Usage:

```
SWEAbstractDataComponent$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>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

`SWEAbstractEncoding` *SWEAbstractEncoding*

Description

SWEAbstractEncoding

SWEAbstractEncoding

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE abstract encoding object

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `SWEAbstractEncoding`

Methods

Public methods:

- `SWEAbstractEncoding$new()`
- `SWEAbstractEncoding$clone()`

Method `new()`**:** Initializes a SWE Nil Values object

Usage:

`SWEAbstractEncoding$new(xml = NULL)`

Arguments:

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

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

Usage:

`SWEAbstractEncoding$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

`SWEAbstractObject` *SWEAbstractObject*

Description

SWEAbstractObject

SWEAbstractObject

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE abstract object

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `SWEAbstractObject`

Methods

Public methods:

- `SWEAbstractObject$new()`
- `SWEAbstractObject$clone()`

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

Usage:

```
SWEAbstractObject$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE,
  value_as_field = FALSE
)
```

Arguments:

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

`element` element

`attrs` attrs

`defaults` defaults

`wrap` wrap

`value_as_field` whether value should be set as field

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

Usage:

```
SWEAbstractObject$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>

`SWEAbstractSimpleComponent` *SWEAbstractSimpleComponent*

Description

SWEAbstractSimpleComponent

SWEAbstractSimpleComponent

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE Abstract simple component

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractSWEIdentifiable` -> `geometa::SWEAbstractDataComponent` -> `SWEAbstractSimpleComponent`

Public fields

`nilValues` nil values

Methods

Public methods:

- `SWEAbstractSimpleComponent$new()`
- `SWEAbstractSimpleComponent$setNilValues()`
- `SWEAbstractSimpleComponent$clone()`

Method `new()`**:** Initializes an object of class SWEAbstractSimpleComponent

Usage:

```
SWEAbstractSimpleComponent$new(
  xml = NULL,
  element = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`element` element

`updatable` updatable

`optional` optional

`definition` definition

Method `setNilValues()`**:** Set nil value and its reason (optional)

Usage:

```
SWEAbstractSimpleComponent$setNilValues(nilValue)
```

Arguments:

`nilValue` value to set as nil Value. object of class `numeric`

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
SWEAbstractSimpleComponent$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

SWEAbstractSWE *SWEAbstractSWE*

Description

SWEAbstractSWE

SWEAbstractSWE

Format

R6Class object.

Value

Object of R6Class for modelling an SWE abstract SWE object

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::SWEAbstractObject -> SWEAbstractSWE

Methods

Public methods:

- SWEAbstractSWE$new()
- SWEAbstractSWE$clone()

Method new()**:** Initializes an object of class SWEAbstractSWE

Usage:

```
SWEAbstractSWE$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  wrap = TRUE,
  value_as_field = FALSE
)
```

Arguments:

xml object of class XMLInternalNode-class from **XML**

element element

attrs attrs

defaults defaults

wrap wrap

value_as_field whether value should be set as field

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
SWEAbstractSWE$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>

`SWEAbstractSWEIdentifiable` *SWEAbstractSWEIdentifiable*

Description

SWEAbstractSWEIdentifiable

SWEAbstractSWEIdentifiable

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE abstract identifiable

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `SWEAbstractSWEIdentifiable`

Public fields

`identifier` identifier

`label` label

`description` description

Methods

Public methods:

- `SWEAbstractSWEIdentifiable$new()`
- `SWEAbstractSWEIdentifiable$setIdentifier()`
- `SWEAbstractSWEIdentifiable$setLabel()`
- `SWEAbstractSWEIdentifiable$setDescription()`
- `SWEAbstractSWEIdentifiable$clone()`

Method `new()`**:** Initializes a SWE Nil Values object

Usage:

```
SWEAbstractSWEIdentifiable$new(
  xml,
  element = element,
  attrs = list(),
  defaults = list(),
  wrap = TRUE,
  value_as_field = TRUE
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`element` element

`attrs` attrs

`defaults` defaults

`wrap` wrap

`value_as_field` value as field?

Method `setIdentifier()`**:** Set identifier

Usage:

```
SWEAbstractSWEIdentifiable$setIdentifier(identifier)
```

Arguments:

`identifier` identifier

Method `setLabel()`**:** Set label

Usage:

```
SWEAbstractSWEIdentifiable$setLabel(label)
```

Arguments:

`label` label

Method `setDescription()`**:** Set description

Usage:

```
SWEAbstractSWEIdentifiable$setDescription(description)
```

Arguments:

`description` description

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
SWEAbstractSWEIdentifiable$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

`SWECategory` *SWECategory*

Description

SWECategory

SWECategory

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE Category

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractSWEIdentifiable` -> `geometa::SWEAbstractDataComponent` -> `geometa::SWEAbstractSimpleComponent` -> `SWECategory`

Public fields

`codeSpace` codeSpace

`constraint` constraint

`value` value

Methods

Public methods:

- `SWECategory$new()`
- `SWECategory$setCodeSpace()`
- `SWECategory$setConstraint()`
- `SWECategory$setValue()`
- `SWECategory$clone()`

Method `new()`**:** Initializes an object of class SWECategory

Usage:

```
SWECategory$new(
  xml = NULL,
  codeSpace = NULL,
  constraint = NULL,
  value = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`codeSpace` codeSpace

`constraint` constraint

`value` value

`updatable` updatable

`optional` optional

`definition` definition

Method `setCodeSpace()`**:** setCodeSpace

Usage:

```
SWECategory$setCodeSpace(codeSpace)
```

Arguments:

`codeSpace` codeSpace

Method `setConstraint()`**:** setConstraint

Usage:

```
SWECategory$setConstraint(constraint)
```

Arguments:

`constraint` constraint

Method `setValue()`**:** setValue

Usage:

```
SWECategory$setValue(value)
```

Arguments:

`value` value

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
SWECategory$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

`SWECategoryRange` *SWECategoryRange*

Description

SWECategoryRange

SWECategoryRange

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE CategoryRange

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractSWEIdentifiable` -> `geometa::SWEAbstractDataComponent` -> `geometa::SWEAbstractSimpleComponent` -> `SWECategoryRange`

Public fields

`codeSpace` codeSpace

`constraint` constraint

`value` value

Methods

Public methods:

- `SWECategoryRange$new()`
- `SWECategoryRange$setCodeSpace()`
- `SWECategoryRange$setConstraint()`
- `SWECategoryRange$setValue()`
- `SWECategoryRange$clone()`

Method `new()`**:** Initializes an object of class SWECategoryRange

Usage:

```
SWECategoryRange$new(
  xml = NULL,
  codeSpace = NULL,
  constraint = NULL,
  value = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`codeSpace` codeSpace

`constraint` constraint

`value` value

`updatable` updatable

`optional` optional

`definition` definition

Method `setCodeSpace()`**:** setCodeSpace

Usage:

```
SWECategoryRange$setCodeSpace(codeSpace)
```

Arguments:

`codeSpace` codeSpace

Method `setConstraint()`**:** setConstraint

Usage:

```
SWECategoryRange$setConstraint(constraint)
```

Arguments:

`constraint` constraint

Method `setValue()`**:** setValue

Usage:

```
SWECategoryRange$setValue(value)
```

Arguments:

`value` value

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
SWECategoryRange$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

`SWECount` *SWECount*

Description

SWECount

SWECount

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE Count

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractSWEIdentifiable` -> `geometa::SWEAbstractDataComponent` -> `geometa::SWEAbstractSimpleComponent` -> `SWECount`

Public fields

`constraint` constraint

`value` value

Methods

Public methods:

- `SWECount$new()`
- `SWECount$setConstraint()`
- `SWECount$setValue()`
- `SWECount$clone()`

Method `new()`**:** Initializes an object of class SWECount

Usage:

```
SWECount$new(
  xml = NULL,
  constraint = NULL,
  value = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`constraint` constraint

`value` value

`updatable` updatable

`optional` optional

`definition` definition

Method `setConstraint()`**:** setConstraint

Usage:

```
SWECount$setConstraint(constraint)
```

Arguments:

`constraint` constraint

Method `setValue()`**:** setValue

Usage:

```
SWECount$setValue(value)
```

Arguments:

`value` value

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
SWECount$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

`SWECountRange` *SWECountRange*

Description

SWECountRange

SWECountRange

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE CountRange

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractSWEIdentifiable` -> `geometa::SWEAbstractDataComponent` -> `geometa::SWEAbstractSimpleComponent` -> `SWECountRange`

Public fields

`constraint` constraint

`value` value

Methods

Public methods:

- `SWECountRange$new()`
- `SWECountRange$setConstraint()`
- `SWECountRange$setValue()`
- `SWECountRange$clone()`

Method `new()`**:** Initializes an object of class SWECountRange

Usage:

```
SWECountRange$new(
  xml = NULL,
  constraint = NULL,
  value = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`constraint` constraint

`value` value

`updatable` updatable

`optional` optional

`definition` definition

Method `setConstraint():` setConstraint

Usage:

```
SWECountRange$setConstraint(constraint)
```

Arguments:

`constraint` constraint

Method `setValue():` setValue

Usage:

```
SWECountRange$setValue(value)
```

Arguments:

`value` value

Method `clone():` The objects of this class are cloneable with this method.

Usage:

```
SWECountRange$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

SWEDataRecord *SWEDataRecord*

Description

SWEDataRecord

SWEDataRecord

Format

R6Class object.

Value

Object of R6Class for modelling an SWE data record

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::SWEAbstractObject -> geometa::SWEAbstractSWE -> geometa::SWEAbstractSWEIdentifiable -> geometa::SWEAbstractDataComponent -> SWEDataRecord

Public fields

`field` field

Methods

Public methods:

- `SWEDataRecord$new()`
- `SWEDataRecord$addField()`
- `SWEDataRecord$delField()`
- `SWEDataRecord$clone()`

Method `new()`**:** Initializes an object of class SWEDataRecord

Usage:

```
SWEDataRecord$new(
  xml = NULL,
  element = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`element` element

`updatable` updatable

`optional` optional

`definition` definition

Method `addField()`**:** Adds field

Usage:

```
SWEDataRecord$addField(field)
```

Arguments:

`field` field

Method `delField()`**:** Deletes field

Usage:

```
SWEDataRecord$delField(field)
```

Arguments:

`field` field

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
SWEDataRecord$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>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

`SWEElement` *SWEElement*

Description

SWEElement

SWEElement

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an GML element

Methods

`new(xml, element, attrs, defaults)` This method is used to instantiate a GML element

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `SWEElement`

Methods

Public methods:

- `SWEElement$new()`
- `SWEElement$decode()`
- `SWEElement$clone()`

Method `new()`: Initializes a generic abstract SWE element

Usage:

```
SWEElement$new(
  xml = NULL,
  element = NULL,
  attrs = list(),
  defaults = list(),
  xmlNamespacePrefix = "SWE"
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`element` element

`attrs` attrs

`defaults` defaults

`xmlNamespacePrefix` XML namespace prefix. Default is `"SWE"`

Method `decode()`: Decodes object from XML

Usage:

```
SWEElement$decode(xml)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
SWEElement$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Note

Class used by geometa internal XML decoder/encoder

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

ISO/TS 19103:2005 Geographic information – Conceptual schema language

---

`SWENilValues` *SWENilValues*

Description

SWENilValues

SWENilValues

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE nil values object

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `SWENilValues`

Public fields

`nilValue` nil value

Methods

Public methods:

- `SWENilValues$new()`
- `SWENilValues$addNilValue()`
- `SWENilValues$clone()`

Method `new()`**:** Initializes a SWE Nil Values object

Usage:

```
SWENilValues$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

Method `addNilValue()`**:** Adds a nil value with a reason

Usage:

```
SWENilValues$addNilValue(value, reason)
```

Arguments:

`value` value

`reason` reason

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
SWENilValues$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

`SWEQuantity` *SWEQuantity*

Description

SWEQuantity

SWEQuantity

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE Quantity

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractSWEIdentifiable` -> `geometa::SWEAbstractDataComponent` -> `geometa::SWEAbstractSimpleComponent` -> `SWEQuantity`

Public fields

`uom` uom

`constraint` constraint

`value` value

Methods

Public methods:

- `SWEQuantity$new()`
- `SWEQuantity$setUom()`
- `SWEQuantity$setConstraint()`
- `SWEQuantity$setValue()`
- `SWEQuantity$clone()`

Method `new()`**:** Initializes an object of class SWEQuantity

Usage:

```
SWEQuantity$new(
  xml = NULL,
  uom = NULL,
  constraint = NULL,
  value = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`uom` uom

`constraint` constraint

`value` value

`updatable` updatable

`optional` optional

`definition` definition

Method `setUom()`**:** setUom

Usage:

```
SWEQuantity$setUom(uom)
```

Arguments:

`uom` uom

Method `setConstraint()`**:** setConstraint

Usage:

```
SWEQuantity$setConstraint(constraint)
```

Arguments:

`constraint` constraint

Method `setValue():` setValue

Usage:

`SWEQuantity$setValue(value)`

Arguments:

`value` value

Method `clone():` The objects of this class are cloneable with this method.

Usage:

`SWEQuantity$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

`SWEQuantityRange` *SWEQuantityRange*

Description

SWEQuantityRange

SWEQuantityRange

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE QuantityRange

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractSWEIdentifiable` -> `geometa::SWEAbstractDataComponent` -> `geometa::SWEAbstractSimpleComponent` -> `SWEQuantityRange`

Public fields

`uom` uom

`constraint` constraint

`value` value

Methods

Public methods:

- `SWEQuantityRange$new()`
- `SWEQuantityRange$setUom()`
- `SWEQuantityRange$setConstraint()`
- `SWEQuantityRange$setValue()`
- `SWEQuantityRange$clone()`

Method `new()`**:** Initializes an object of class SWEQuantityRange

Usage:

```
SWEQuantityRange$new(
  xml = NULL,
  uom = NULL,
  constraint = NULL,
  value = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`uom` uom

`constraint` constraint

`value` value

`updatable` updatable

`optional` optional

`definition` definition

Method `setUom()`**:** setUom

Usage:

```
SWEQuantityRange$setUom(uom)
```

Arguments:

`uom` uom

Method `setConstraint()`**:** setConstraint

Usage:

```
SWEQuantityRange$setConstraint(constraint)
```

Arguments:

`constraint` constraint

Method `setValue()`: setValue

Usage:

`SWEQuantityRange$setValue(value)`

Arguments:

`value` value

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

`SWEQuantityRange$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

`SWEText` *SWEText*

Description

SWEText

SWEText

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE Text

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractSWEIdentifiable` -> `geometa::SWEAbstractDataComponent` -> `geometa::SWEAbstractSimpleComponent` -> `SWEText`

Public fields

`constraint` constraint

`value` value

Methods

Public methods:

- `SWEText$new()`
- `SWEText$setConstraint()`
- `SWEText$setValue()`
- `SWEText$clone()`

Method `new()`**:** Initializes an object of class SWEText

Usage:

```
SWEText$new(
  xml = NULL,
  constraint = NULL,
  value = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`constraint` constraint

`value` value

`updatable` updatable

`optional` optional

`definition` definition

Method `setConstraint()`**:** setConstraint

Usage:

```
SWEText$setConstraint(constraint)
```

Arguments:

`constraint` constraint

Method `setValue()`**:** setValue

Usage:

```
SWEText$setValue(value)
```

Arguments:

`value` value

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
SWEText$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

SWETextEncoding *SWETextEncoding*

Description

SWETextEncoding

SWETextEncoding

Format

R6Class object.

Value

Object of R6Class for modelling an SWE text encoding object

Super classes

geometa::geometaLogger -> geometa::ISOAbstractObject -> geometa::SWEAbstractObject -> geometa::SWEAbstractSWE -> geometa::SWEAbstractEncoding -> SWETextEncoding

Methods

Public methods:

- SWETextEncoding$new()
- SWETextEncoding$clone()

Method new()**:** Initializes a SWE Text Encoding element

Usage:

```
SWETextEncoding$new(
  xml = NULL,
  collapseWhiteSpaces = TRUE,
  decimalSeparator = ".",
  tokenSeparator = NULL,
  blockSeparator = NULL
)
```

Arguments:

xml object of class XMLInternalNode-class from **XML** `collapseWhiteSpaces` Indicates whether white spaces (i.e. space, tab, CR, LF) should be collapsed with separators when parsing the data stream. Default is `TRUE`

`decimalSeparator` Character used as the decimal separator. Default is `TRUE`

`tokenSeparator` Character sequence used as the token separator (i.e. between two successive values). Required

`blockSeparator` Character sequence used as the block separator (i.e. between two successive blocks in the data set. The end of a block is reached once all values from the data tree have been encoded once). Required

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
SWETextEncoding$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

`SWETime` *SWETime*

Description

SWETime

SWETime

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE Time

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractSWEIdentifiable` -> `geometa::SWEAbstractDataComponent` -> `geometa::SWEAbstractSimpleComponent` -> `SWETime`

Public fields

`uom` uom

`constraint` constraint

`value` value

Methods

Public methods:

- `SWETime$new()`
- `SWETime$setUom()`
- `SWETime$setConstraint()`
- `SWETime$setValue()`
- `SWETime$clone()`

Method `new()`**:** Initializes an object of class SWETime

Usage:

```
SWETime$new(
  xml = NULL,
  uom = NULL,
  constraint = NULL,
  value = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`uom` uom

`constraint` constraint

`value` value

`updatable` updatable

`optional` optional

`definition` definition

Method `setUom()`**:** setUom

Usage:

```
SWETime$setUom(uom)
```

Arguments:

`uom` uom

Method `setConstraint()`**:** setConstraint

Usage:

```
SWETime$setConstraint(constraint)
```

Arguments:

`constraint` constraint

Method `setValue():` setValue

Usage:

```
SWETime$setValue(value)
```

Arguments:

`value` value

Method `clone():` The objects of this class are cloneable with this method.

Usage:

```
SWETime$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

`SWETimeRange` *SWETimeRange*

Description

SWETimeRange

SWETimeRange

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE Time Range

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractSWEIdentifiable` -> `geometa::SWEAbstractDataComponent` -> `geometa::SWEAbstractSimpleComponent` -> `SWETimeRange`

Public fields

`uom` uom

`constraint` constraint

`value` value

Methods

Public methods:

- `SWETimeRange$new()`
- `SWETimeRange$setUom()`
- `SWETimeRange$setConstraint()`
- `SWETimeRange$setValue()`
- `SWETimeRange$clone()`

Method `new()`: Initializes an object of class SWETimeRange

Usage:

```
SWETimeRange$new(
  xml = NULL,
  uom = NULL,
  constraint = NULL,
  start = NULL,
  end = NULL,
  updatable = NULL,
  optional = FALSE,
  definition = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`uom` uom

`constraint` constraint

`start` start time

`end` end time

`updatable` updatable

`optional` optional

`definition` definition

Method `setUom()`: setUom

Usage:

```
SWETimeRange$setUom(uom)
```

Arguments:

`uom` uom

Method `setConstraint()`: setConstraint

Usage:

`SWETimeRange$setConstraint(constraint)`

Arguments:

`constraint` constraint

Method `setValue()`**:** setValue

Usage:

`SWETimeRange$setValue(start, end)`

Arguments:

`start` start time

`end` end time

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

`SWETimeRange$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

---

`SWEXMLEncoding` *SWEXMLEncoding*

Description

SWEXMLEncoding

SWEXMLEncoding

Format

`R6Class` object.

Value

Object of `R6Class` for modelling an SWE XML encoding object

Super classes

`geometa::geometaLogger` -> `geometa::ISOAbstractObject` -> `geometa::SWEAbstractObject` -> `geometa::SWEAbstractSWE` -> `geometa::SWEAbstractEncoding` -> `SWEXMLEncoding`

Methods

Public methods:

- `SWEXMLEncoding$new()`
- `SWEXMLEncoding$clone()`

Method `new()`**:** Initializes a SWE XML Encoding element

Usage:

```
SWEXMLEncoding$new(xml = NULL)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
SWEXMLEncoding$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

References

SWE Common Data Model Encoding Standard. https://www.ogc.org/standards/swecommon

Index