MI - Fimex

Generated by Doxygen 1.6.3

Tue Nov 8 12:57:09 2011

Contents

Chapter 1

Fimex User Documentation

Fimex is a the File Interpolation, Manipulation and EXtraction library for gridded geospatial data. It converts between different, extensible dataformats (currently netcdf, grib1/2 and felt). It enables you to change the projection and interpolation of scalar and vector grids. It makes it possible subset the gridded data and to extract only parts of the files.

Fimex can be used as library called *Fimex* and a command-line program called *fimex*, which gives access to most but not all functions of the library.

Fimex is build around the Common Data Model version 1 developped by Unidata and uses a describes data using the CF-Convention http://cf-pcmdi.llnl.gov/documents/cf-conventions/1.0/cf-conventions.html . Knowledge of that convention is not required, but will help understanding the config files needed for conversion.

The API of Fimex as included in this document is not stable yet and can change without warning. The setup-files are considered to be mostly stable. The fimex-program can thus savely be used. If you want to use the API, please contact me.

1.1 Setup Files

Detailed information on the differnt configuration files can be found at:

- fimex Program Options
- Configuration files for felt reader
- ncml Configuration
- wdbReaderDoc
- gribWriter Configuration
- netcdfWriter Configuration
- quality-extraction Configuration

Chapter 2

fimex Program Options

2.1 fimex Program Options

fimex is a command-line program. It has the following options:

```
usage: fimex --input.file  FILENAME [--input.type  INPUT_TYPE]
             --output.file FILENAME [--output.type OUTPUT_TYPE]
             [--input.config CFGFILENAME] [--output.config CFGFILENAME]
             [--extract....]
             [--interpolate....]
             [--timeInterpolate....]

Generic options:
  -h [ --help ]                   help message
  --version                       program version
  --debug                         debug program
  --print-options                 print all options
  -c [ --config ] arg (=fimex.cfg) configuration file

Configurational options:
  --input.file arg                        input file
  --input.type arg                        filetype of input file, e.g. nc, nc4,
                                          ncml, felt, grib1, grib2, wdb
  --input.config arg                      non-standard input configuration
  --input.printNcML                       print NcML description of input file
  --input.printCS                         print CoordinateSystems of input file
  --output.file arg                       output file
  --output.type arg                       filetype of output file, e.g. nc,
                                          nc4, grib1, grib2
  --output.config arg                     non-standard output configuration
  --extract.removeVariable arg            remove variables
  --extract.selectVariables arg           select only those variables
  --extract.reduceDimension.name arg      name of a dimension to reduce
  --extract.reduceDimension.start arg     start position of the dimension to
                                          reduce (>=0)
  --extract.reduceDimension.end arg       end position of the dimension to
                                          reduce
  --extract.reduceTime.start arg          start-time as iso-string
  --extract.reduceTime.end arg            end-time by iso-string
  --extract.reduceVerticalAxis.unit arg   unit of vertical axis to reduce
  --extract.reduceVerticalAxis.start arg  start value of vertical axis
  --extract.reduceVerticalAxis.end arg    end value of the vertical axis
  --extract.reduceToBoundingBox.south arg geographical bounding-box in degree
  --extract.reduceToBoundingBox.north arg geographical bounding-box in degree
  --extract.reduceToBoundingBox.east arg  geographical bounding-box in degree
  --extract.reduceToBoundingBox.west arg  geographical bounding-box in degree
  --extract.printNcML                     print NcML description of extractor
  --extract.printCS                       print CoordinateSystems of extractor
  --qualityExtract.autoConfString arg     configure the quality-assignment
                                          using CF-1.3 status-flag
  --qualityExtract.config arg             configure the quality-assignment with
                                          a xml-config file
  --qualityExtract.printNcML              print NcML description of extractor
  --qualityExtract.printCS                print CoordinateSystems of extractor
  --interpolate.projString arg            proj4 input string describing the new
                                          projection
  --interpolate.method arg                interpolation method, one of
                                          nearestneighbor, bilinear, bicubic,
                                          coord_nearestneighbor, coord_kdtree,
                                          forward_max, forward_mean,
                                          forward_median or forward_sum
  --interpolate.xAxisValues arg           string with values on x-Axis, use ...
                                          to continue, i.e. 10.5,11,...,29.5,
                                          see Fimex::SpatialAxisSpec for full
                                          definition
  --interpolate.yAxisValues arg           string with values on x-Axis, use ...
                                          to continue, i.e. 10.5,11,...,29.5,
```

```
                                    see Fimex::SpatialAxisSpec for full
                                    definition
  --interpolate.xAxisUnit arg       unit of x-Axis given as udunits
                                    string, i.e. m or degrees_east
  --interpolate.yAxisUnit arg       unit of y-Axis given as udunits
                                    string, i.e. m or degrees_north
  --interpolate.latitudeName arg    name for auto-generated projection
                                    coordinate latitude
  --interpolate.longitudeName arg   name for auto-generated projection
                                    coordinate longitude
  --interpolate.preprocess arg      add a 2d preprocess to before the
                                    interpolation, i.e.
                                    "fill2d(critx,cor,maxLoop)"
  --interpolate.printNcML           print NcML description of
                                    interpolator
  --interpolate.printCS             print CoordinateSystems of
                                    interpolator
  --timeInterpolate.timeSpec arg    specification of times to interpolate
                                    to, see Fimex::TimeSpec for a full
                                    definition
  --timeInterpolate.printNcML       print NcML description of
                                    timeInterpolator
  --timeInterpolate.printCS         print CoordinateSystems of
                                    timeInterpolator
  --ncml.config                     modify/configure with ncml-file
  --ncml.printNcML                  print NcML description after
                                    ncml-configuration
  --ncml.printCS                    print CoordinateSystems after
                                    ncml-configuration
```

All the configurational options can be configured using a configuration file which is supplied using the --config option. All command line options (CLO) will overwrite the config-file. As a rule of thump, use the CLO for testing and use the config-file for productive usage. The CLOs will be further explained in fimex Setup File.

2.1.1 fimex Setup File

The *SpatialAxisSpec* used in xAxisValues or yAxisValues for the spatial interpolation should be formatted as explained in detail in MetNoFimex::SpatialAxisSpec. It allows also autotuning to the orignal data-values.

The *TimeSpec* string used for the timeInterpolate should be formatted as explained in detail in MetNoFimex::TimeSpec.

Chapter 3

Configuration files for felt reader

The xml configuration files are defined by the *felt2nc_variables.dtd* definition. Since part of this configuration are quite stable, e.g. the axes (time, level, lat, lon, x, y), other parts change, e.g. the variables to translate change very often. It is therefore useful to split the variables from the rest of the configuration via *xinclude*

When writing a new configuration for a new set of felt-files, usually from a new model, it is wise to group the configuration by

1. time resolution, i.e. one config for 3hourly files, one config for hourly files
2. spatial resolution: fimex doesn't allow different spatial resolutions, but some models use coarser resoluton for higher levels
3. vertical levels: it is difficult to have the same parameter with sigma levels and with height in m

Grouping can be done in two ways, the first one being faster in operation, the second is easier to configure/change consistently:

1. write different configuration-files for each group of parameter, stating the parameter as well as possible.
2. write one configuration-file for all parameter, keeping the parameters as variable as possible. Use a preprocess-step to extract each group. Use e.g. *nyfelt* or *felt2felt* as preprocessor

By default, all data is read as *type="short"* data with a scaling factor. While felt allows for one scaling factor for each timestep, height and parameter, the CDM allows only for one scaling factor per parameter. When the scaling factor changes withing height or timestep, *fimex* will fail to read the data as short. It is therefore useful to read data as *type="float"*, which will automatically expand the scaling factor. If the resulting file is to big, it is possible to convert to short with one scaling factor and offset using the netcdfWriter Configuration.

Before running fimex with a new felt configuration, make sure the file is valid, e.g. with

```
xmllint --valid --noout felt2nc_config.xml
```

Unfortuneatly, xinclude and validation don't play well together, since usual validation happens before the inclusion of external parts. xmllint uses special options to fix those problem:

```
xmllint --xinclude --postvalid --noout felt2nc_config.xml
```

Below follows a complete felt-configuration.

Chapter 4

ncml Configuration

4.1 ncml Configuration

Unidata's NetCDF Markup Language (NcML) as described in http://www.unidata.ucar.edu/software/netcdf/ncml/ gives the opportunity to change all information written in the CDM. With the --ncml.config option, the CDM will be configured immediately after reading a file. It is also possilbe to read in a ncml file with the --input.file=xxx.ncml option. In this case, the real data must be linked with the 'location' markup.

Input-files can and should be validated against the included ncml-2.2.xsd.

Warning

not all features are supported in the current implementation. Missing features are: aggregation, changing dimension sizes, unlimited dimensions, adding/changing values of variables, groups

Chapter 5

wdb Reader Configuration

5.1 wdb Reader Configuration

This feature is still alpha-quality and will change!

Fimex is able to read data from instances of the WDB (http://wdb.sourceforge.net) by using the WdbCDMReader (or on the command line: input.type=wdb). The usual 'file'-location is in that case reinterpreted as wdb-connetction string, e.g. 'dbHost=proffdb-devel.met.no;dbPort=5432;dbName=wdb;dbUser=proffread;wciUser=proffwrite' would connect to the host proffdb-devel on port 5432 and the databse wdb with the user proffread. Additonal options:

- refTime use exactly this reference time
- provider use this provider
- place use this place

Example:

```
fimex --input.file='dbHost=proffdb-devel.met.no;dbPort=5432;dbName=wdb;dbUser=pro
      ffread;refTime=20110211T000000' --input.type=wdb --input.config=/home/heikok/Prog
      ramme/MetSis/Fimex/share/etc/proffdb_config.xml --output.file=test.nc --output.ty
      pe=nc4 --extract.reduceDimension.name=time --extract.reduceDimension.start=1 --ex
      tract.reduceDimension.end=3
```

Most of these options, and additional one adding correct metadata can be given in the config file, e.g. in proffdb_config.xml :

Chapter 6

quality-extraction Configuration

6.1 quality-extraction Configuration

Warning

The quality-extraction is still in a very early stage of development. The configuration and the outcome is very likely to change in further developments. Any feedback is strongly welcome.

In cases where the data should be extracted if certain conditions (qualities) apply, i.e. the status-flag indicates a properly working instrument, or the sea-surface-temperature is above 300K, the MetNoFimex::CDMQualityExtractor allows to add these rules. The cdmQualityConfig.xml file as shown above gives an example of such an configuration.

- The variable "bla" will only be set, if "blub" has integer-values between 1 and 6.
- The variable "air_temperature" will only be extracted for an "altitude" above 1000. The value 1000 is the actual data value in the variable "altitude" without any scaling or unit-conversion applied.

The following use-values can be selected:

- `all` select all valid values (within valid_max, valid_min or valid_range, without _FillValue)
- `highest` the highest numerical value found in the data-slice which is valid
- `lowest` the lowest numerical value fond in the data-slice which is valid
- `max:xxx.x` all valid-values below or equal xxx.x
- `min:xxx.x` all valid values above or equal xxx.x

All values which do not match the quality-criteria will be set to the _FillValue of the variable.

Chapter 7

gribWriter Configuration

7.1 gribWriter Configuration

Chapter 8

netcdfWriter Configuration

8.1 netcdfWriter Configuration

The netcdfWriterConfig gives the opportunity to set some features explicit only for netcdf-files, i.e. file-format (netcdf3/4) or compression.

It is also possible to add an ncml Configuration to the output to change the internal structure just before writing.

It is also possible ot change units including all value in the netcdfWriterDoc. Changing the units in the ncmlConfiguration would change the attribute value only, but not the data.

The CDM resembles a netcdf datastructure. In general, there is no need to use a configuration for this writer, but it might be useful in the following cases:

- Output-files are to big, and a change of datatype i.e. from float to short is desired
- Different attribute are required for special usages, but the input-configuration of the reader shouldn't been changed.
- Different variable or dimension names are required for special usages.

Chapter 9

Deprecated List

Member DEPRECATED(int fimexHasNetcdf()) use fimexHas(fileType)

use fimexHas(fileType)

use fimexHas(fileType)

Member MetNoFimex::CDM::DEPRECATED(AttrVec getProjection(std::string varName) const)
use the getProjectionOf() method

Member MetNoFimex::CDMFileReaderFactory::create(int fileType, const std::string &fileName, const std::string &co
use create(int fileType, const **std::string**& fileName, const XMLInput& configXML, const std::vector<std::string>& args = std::vector<std::string>())

Member MetNoFimex::CDMFileReaderFactory::create(int fileType, const std::string &fileName, const XMLInput &co
use create(int fileType, const **std::string**& fileName, const XMLInput& configXML, const std::vector<std::string>& args = std::vector<std::string>())

Member MetNoFimex::CDMFileReaderFactory::create(const std::string &fileType, const std::string &fileName, const
use create(const **std::string**& fileType, const **std::string**& fileName, const XMLInput& configXML, const std::vector<std::string>& args = std::vector<std::string>())

Member MetNoFimex::CDMInterpolator::DEPRECATED(virtual void changeProjection(int method, const std::string
use version changeProjection(int method, const std::string& proj_input, const std::vector<double>& out_x_axis, const std::vector<double>& out_y_axis, const std::string& out_x_axis_unit, const std::string& out_y_axis_unit)

Member MetNoFimex::DEPRECATED(std::string attributesToProjString(const std::vector< CDMAttribute > &attrs
use Projection::create() with Projection::getProj4String() instead

Member MetNoFimex::DEPRECATED(std::vector< CDMAttribute > projStringToAttributes(std::string projStr))
use Projection::createByProj4() and Projection::getParameters()

Chapter 10

Namespace Index

10.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 11

Class Index

11.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 12

Class Index

12.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 13

File Index

13.1 File List

Here is a list of all files with brief descriptions:

Chapter 14

Namespace Documentation

14.1 felt Namespace Reference

Classes

- class FeltField
- class FeltFile
- class FeltGridDefinition

Typedefs

- typedef short int word
  A felt block "word" - 2 bytes.
- typedef boost::shared_ptr< FeltGridDefinition > FeltGridDefinitionPtr

Functions

- bool isUndefined (word w)
- boost::array< float, 6 > gridParameters (int gridType, int xNum, int yNum, int a, int b, int c, int d, const **std::vector**< short int > &extraData)
- **std::string** gridParametersToProjDefinition (int gridType, const boost::array< float, 6 > &gridPars)
- **std::ostream** & contentSummary (**std::ostream** &out, const FeltGridDefinition &grid)
- template<typename T >
  T get (word w)
- boost::posix_time::ptime parseTime (const word *data)
- boost::posix_time::ptime parseTimeNoThrow (const word *data)

Variables

- const size_t blockWords = 1024
- const size_t blockSize = blockWords * sizeof(word)
- const size_t offsetToContentDefinition = 2
- const double PI = 3.14159265358979323846264338327 95
- const double EARTH_RADIUS = 6371000.

14.1.1 Typedef Documentation

14.1.1.1 typedef boost::shared_ptr<FeltGridDefinition> felt::FeltGridDefinitionPtr

14.1.1.2 typedef short int felt::word

A felt block "word" - 2 bytes.

14.1.2 Function Documentation

14.1.2.1 std::ostream& felt::contentSummary (std::ostream & *out*, const FeltGridDefinition & *grid*)

14.1.2.2 template<typename T > T felt::get (word *w*) [inline]

14.1.2.3 boost::array<float, 6> felt::gridParameters (int *gridType*, int *xNum*, int *yNum*, int *a*, int *b*, int *c*, int *d*, const std::vector< short int > & *extraData*)

retrieve the 6 gridparameters from the felt-data

Parameters

gridType id of the grid, (header[8] < 1000) ? header[8] : (int) header[8] / 1000

xNum number of points in x-direction, header[9]

yNum number of points in x-direction, header[10]

a used for different depending on gridType, header[14]

b used for different depending on gridType, header[15]

c used for different depending on gridType, header[16]

d used for different depending on gridType, header[17]

extraData data at the end of the data-region, used for high resolution information (header[8] < 1000) ? 0 : header[8] % 1000

14.1.2.4 std::string felt::gridParametersToProjDefinition (int *gridType*, const boost::array< float, 6 > & *gridPars*)

convert the libmi-gridparameters to proj4 strings

Parameters

gridType type defining the projection (1..6)

gridPars array containing libmi's six gridparameters

Returns

proj.4 string

14.1.2.5 bool felt::isUndefined (word *w*) [inline]

Is the given word a "missing" value? This means that there exist no valid data for the given point

14.1.2.6 boost::posix_time::ptime felt::parseTime (const word * *data*)

14.1.2.7 boost::posix_time::ptime felt::parseTimeNoThrow (const word * *data*)

14.1.3 Variable Documentation

14.1.3.1 const size_t felt::blockSize = blockWords * sizeof(word)

14.1.3.2 const size_t felt::blockWords = 1024

14.1.3.3 const double felt::EARTH_RADIUS = 6371000.

earth-radius in m, taken from libmi rearth.f

14.1.3.4 const size_t felt::offsetToContentDefinition = 2

14.1.3.5 const double felt::PI = 3.14159265358979323846264338327 95

mathematical pi

14.2 MetNoFelt Namespace Reference

Classes

- class Felt_Array2

  A collection of FeltFields to build a 4-dimensional data array for 1-parameter and 1-vertical coordinate.

- class Felt_File2

  Felt File access.

- class Felt_File_Error
- struct LevelPairLess
- class FeltParameters

Typedefs

- typedef **std::pair**< short, short > LevelPair

Functions

- **std::string** getProjString (int gridType, const boost::array< float, 6 > &gridParameters)
- const int ANY_VALUE ()
- const **std::string** & UNDEFINED ()
- const boost::array< short, 16 > & ANY_ARRAY ()
- const boost::array< short, 20 > & ANY_ARRAY20 ()

14.2.1 Typedef Documentation

14.2.1.1 typedef std::pair<short, short> MetNoFelt::LevelPair

a pair with two level values

14.2.2 Function Documentation

14.2.2.1 const boost::array<short, 16>& MetNoFelt::ANY_ARRAY ()

14.2.2.2 const boost::array<short, 20>& MetNoFelt::ANY_ARRAY20 ()

14.2.2.3 const int MetNoFelt::ANY_VALUE () `[inline]`

14.2.2.4 std::string MetNoFelt::getProjString (int *gridType*, const boost::array< float, 6 > & *gridParameters*)

14.2.2.5 const std::string& MetNoFelt::UNDEFINED ()

14.3 MetNoFimex Namespace Reference

Classes

- class C_CDMReader
- class CachedForwardInterpolation
- class CachedInterpolationInterface
- class CachedInterpolation
- class CachedVectorReprojection
- class CDM

  Data structure of the Common Data Model.

- class CDMAttribute
- class CDMDimension
- class CDMException
- class CDMExtractor
- class CDMFileReaderFactory
- class InterpolatorProcess2d
- class InterpolatorFill2d
- class InterpolatorCreepFill2d
- class CDMInterpolator
- class CDMNamedEntity
- struct CDMNameCompare
- class CDMNameEqual
- class CDMNameEqualPtr
- class CDMPressureConversions
- class CDMQualityExtractor

  Extract data with defined quality status.

- class CDMReader

  Basic interface for CDM reading and manipulation classes.

- class CDMTimeInterpolator
- class CDMVariable
- class CDMVerticalInterpolator

  Interpolation of vertical layers.

- class CDMWriter
- class CoordinateSystemSliceBuilder
- class AlbersConicalEqualAreaProjection
- class AzimuthalEquidistantProjection
- class CoordinateAxis
- class CoordinateSystem
- struct CompleteCoordinateSystemForComparator
- class LambertAzimuthalEqualAreaProjection
- class LambertConformalConicProjection
- class LambertCylindricalEqualAreaProjection
- class LatitudeLongitudeProjection
- class MercatorProjection
- class OrthographicProjection

- class PolarStereographicProjection
- class Projection
- class ProjectionImpl
- class RotatedLatitudeLongitudeProjection
- class StereographicProjection
- class TransverseMercatorProjection
- class UnknownToFgdcProjection
- class VerticalPerspectiveProjection
- class Data
- class DataTypeChanger
- class FeltCDMReader2
- class GribApiCDMWriter
- class GribApiCDMWriter_Impl1
- class GribApiCDMWriter_Impl2
- class GribApiCDMWriter_ImplAbstract
- class GribCDMReader
- class GribFileMessage
- class GribFileMessageEqualTime

  Functor to find Messages with equal time.

- class GribFileMessageEqualLevelTime

  Functor to find messages with equal level and time.

- class GribFileIndex
- class GridDefinition
- class Logger
- class MetGmCDMReader
- class MetGmCDMWriter
- class NcmlCDMReader
- class NetCDF_CDMReader
- class NetCDF_CDMWriter
- class Null_CDMWriter
- class ReplaceStringObject
- class ReplaceStringTimeObject
- class SliceBuilder
- class SpatialAxisSpec
- class TimeLevelDataSliceFetcher

  read a slice of a given time/level combination from a cdmReader

- class TimeSpec
- class FimexTime
- class TimeUnit
- class UnitException
- class Units
- struct staticCast
- class ScaleValue
- class ChangeMissingValue
- struct SharedArrayConstCastDeleter
- class WdbCDMReader
- class XMLDoc

- class XMLInput
- class XMLInputFile
- class XMLInputString
- class XMLInputURL

Typedefs

- typedef boost::shared_ptr< Logger > LoggerPtr
- typedef long epoch_seconds
- typedef boost::shared_ptr< xmlXPathObject > XPathObjPtr

Enumerations

- enum CDMDataType {
  CDM_NAT = 0, CDM_CHAR, CDM_SHORT, CDM_INT,
  CDM_FLOAT, CDM_DOUBLE, CDM_STRING, CDM_UCHAR,
  CDM_USHORT, CDM_UINT, CDM_INT64, CDM_UINT64 }

Functions

- DEPRECATED (**std::vector**< CDMAttribute > projStringToAttributes(**std::string** projStr))
  convert a proj4 string to a list of CDMAttributes usable for CF-1.0 projection variable
- DEPRECATED (**std::string** attributesToProjString(const **std::vector**< CDMAttribute > &attrs))
  convert attributes of a projection-variable to a projString
- CDMDataType string2datatype (const **std::string** &s)
  translate float/string/... to the appropriate CDMDataType
- **std::string** datatype2string (CDMDataType type)
- boost::posix_time::ptime getUniqueForecastReferenceTime (boost::shared_ptr< CDMReader > reader)
- **std::ostream** & operator<< (**std::ostream** &out, CoordinateAxis ca)
- **std::ostream** & operator<< (**std::ostream** &out, CoordinateAxis::AxisType t)
- **std::ostream** & operator<< (**std::ostream** &out, const CoordinateSystem &p)
- **std::vector**< boost::shared_ptr< const CoordinateSystem > > listCoordinateSystems (const CDM &cdm)
- **std::ostream** & operator<< (**std::ostream** &out, const Projection &proj)
- boost::shared_ptr< Data > createData (CDMDataType datatype, size_t length, double val=0)
  create a Data-pointer of the datatype
- boost::shared_ptr< Data > createData (size_t length, boost::shared_array< double > array)
  create a Data-pointer of type CDM_DOUBLE
- boost::shared_ptr< Data > createData (size_t length, boost::shared_array< float > array)
  create a Data-pointer of type CDM_FLOAT
- boost::shared_ptr< Data > createData (size_t length, boost::shared_array< int > array)

create a Data-pointer of type CDM_INT

- boost::shared_ptr< Data > createData (size_t length, boost::shared_array< short > array)

  create a Data-pointer of type CDM_SHORT

- boost::shared_ptr< Data > createData (size_t length, boost::shared_array< char > array)

  create a Data-pointer of type CDM_CHAR

- boost::shared_ptr< Data > createData (size_t length, boost::shared_array< unsigned int > array)

  create a Data-pointer of type CDM_UINT

- boost::shared_ptr< Data > createData (size_t length, boost::shared_array< long long > array)

  create a Data-pointer of type CDM_INT64

- boost::shared_ptr< Data > createData (size_t length, boost::shared_array< unsigned long long > array)

  create a Data-pointer of type CDM_UINT64

- boost::shared_ptr< Data > createData (size_t length, boost::shared_array< unsigned short > array)

  create a Data-pointer of type CDM_USHORT

- boost::shared_ptr< Data > createData (size_t length, boost::shared_array< unsigned char > array)

  create a Data-pointer of type CDM_UCHAR

- template<class InputIterator >
  boost::shared_ptr< Data > createData (CDMDataType datatype, InputIterator first, InputIterator last)

  create a Data-pointer of the datatype and fill with the data from the iterator

- boost::shared_ptr< Data > createDataSlice (CDMDataType datatype, const Data &data, size_t dataStartPos, size_t dataSize)

  create a one-dimensional dataslice from another Data object

- **std::ostream** & operator<< (**std::ostream** &os, const GribFileMessage &gfm)

  outputstream for a GribFileMessage

- **std::ostream** & operator<< (**std::ostream** &os, const GribFileIndex &gfm)

  outputstream for a GribFileIndex

- GridDefinition::Orientation gribGetGridOrientation (boost::shared_ptr< grib_handle > gh)
- Logger::LogLevel defaultLogLevel ()
- void defaultLogLevel (Logger::LogLevel)
- LoggerPtr getLogger (const **std::string** &className)
- **std::ostream** & operator<< (**std::ostream** &out, const FimexTime &fTime)

  minimum FimexTime

- FimexTime string2FimexTime (const **std::string** &str) throw (CDMException)
- void handleUdUnitError (int unitErrCode, const **std::string** &message="") throw (UnitException)
- int round (double num)

- **std::string** trim (const **std::string** &str)
- template<class InputIterator >
  std::string join (InputIterator start, InputIterator end, **std::string** delim=",")
- template<typename InputIterator >
  std::pair< typename **std::iterator_traits**< InputIterator >::difference_type, typename **std::iterator_traits**< InputIterator >::difference_type > find_closest_distinct_elements (InputIterator start, InputIterator end, double x)
- template<typename InputIterator >
  std::pair< typename **std::iterator_traits**< InputIterator >::difference_type, typename **std::iterator_traits**< InputIterator >::difference_type > find_closest_neighbor_distinct_elements (InputIterator start, InputIterator end, double x)
- template<class InputIterator >
  std::string joinPtr (InputIterator start, InputIterator end, **std::string** delim=",")
- **std::vector**< **std::string** > tokenize (const **std::string** &str, const **std::string** &delimiters=" ")
- **std::string** string2lowerCase (const **std::string** &str)
- template<typename T >
  std::string type2string (T in)
- template<>
  std::string type2string< double > (double in)
- template<typename T >
  T string2type (**std::string** s)
- epoch_seconds posixTime2epochTime (const boost::posix_time::ptime &time)
- template<typename T >
  std::vector< T > tokenizeDotted (const **std::string** &str, const **std::string** &delimiter=",") throw (CDMException)
- template<typename T >
  boost::shared_array< const T > makeSharedArrayConst (const boost::shared_array< T > &sa)
- **std::string** getXmlProp (const xmlNodePtr node, const **std::string** &attrName)
- **std::string** getXmlName (const xmlNodePtr node)
- **std::string** getXmlContent (const xmlNodePtr node)

  get all text-contents of the node or underlying nodes

14.3.1 Typedef Documentation

14.3.1.1 typedef long MetNoFimex::epoch_seconds

14.3.1.2 typedef boost::shared_ptr<Logger> MetNoFimex::LoggerPtr

14.3.1.3 typedef boost::shared_ptr<xmlXPathObject> MetNoFimex::XPathObjPtr

14.3.2 Enumeration Type Documentation

14.3.2.1 enum MetNoFimex::CDMDataType

Be aware that the CDM_CHAR datatype maps to NC_BYTE, while the CDM_SHORT maps to NC_-CHAR

Enumerator:

CDM_NAT

CDM_CHAR

CDM_SHORT

CDM_INT

CDM_FLOAT

CDM_DOUBLE

CDM_STRING

CDM_UCHAR

CDM_USHORT

CDM_UINT

CDM_INT64

CDM_UINT64

14.3.3 Function Documentation

14.3.3.1 template<class InputIterator > boost::shared_ptr< Data > MetNoFimex::createData (CDMDataType *datatype*, InputIterator *first*, InputIterator *last*) [inline]

create a Data-pointer of the datatype and fill with the data from the iterator

Parameters

datatype

first start of container containing the data to fill the array with

last end (excluded) of the container containing the data to fill the array with

Returns

Base-Class ptr of the DataImpl belonging to the datatype

References CDM_CHAR, CDM_DOUBLE, CDM_FLOAT, CDM_INT, CDM_INT64, CDM_NAT, CDM_SHORT, CDM_UCHAR, CDM_UINT, CDM_UINT64, CDM_USHORT, std::copy(), createData(), and std::distance().

14.3.3.2 boost::shared_ptr<Data> MetNoFimex::createData (size_t *length*, boost::shared_array< unsigned char > *array*)

create a Data-pointer of type CDM_UCHAR

Parameters

size_t length of the data array

array the data array

Returns

Base-Class ptr of the DataImpl belonging to the datatype

14.3.3.3 boost::shared_ptr<Data> MetNoFimex::createData (size_t *length*, boost::shared_array< unsigned short > *array*)

create a Data-pointer of type CDM_USHORT

Parameters

size_t length of the data array

array the data array

Returns

Base-Class ptr of the DataImpl belonging to the datatype

14.3.3.4 boost::shared_ptr<Data> MetNoFimex::createData (size_t *length*, boost::shared_array< unsigned long long > *array*)

create a Data-pointer of type CDM_UINT64

Parameters

size_t length of the data array

array the data array

Returns

Base-Class ptr of the DataImpl belonging to the datatype

14.3.3.5 boost::shared_ptr<Data> MetNoFimex::createData (size_t *length*, boost::shared_array< long long > *array*)

create a Data-pointer of type CDM_INT64

Parameters

size_t length of the data array

array the data array

Returns

Base-Class ptr of the DataImpl belonging to the datatype

14.3.3.6 boost::shared_ptr<Data> MetNoFimex::createData (size_t *length*, boost::shared_array< unsigned int > *array*)

create a Data-pointer of type CDM_UINT

Parameters

size_t length of the data array

array the data array

Returns

Base-Class ptr of the DataImpl belonging to the datatype

14.3.3.7 boost::shared_ptr<Data> MetNoFimex::createData (size_t *length*, boost::shared_array< char > *array*)

create a Data-pointer of type CDM_CHAR

Parameters

size_t length of the data array

array the data array

Returns

Base-Class ptr of the DataImpl belonging to the datatype

14.3.3.8 boost::shared_ptr<Data> MetNoFimex::createData (size_t *length*, boost::shared_array< short > *array*)

create a Data-pointer of type CDM_SHORT

Parameters

size_t length of the data array

array the data array

Returns

Base-Class ptr of the DataImpl belonging to the datatype

14.3.3.9 boost::shared_ptr<Data> MetNoFimex::createData (size_t *length*, boost::shared_array< int > *array*)

create a Data-pointer of type CDM_INT

Parameters

size_t length of the data array

array the data array

Returns

Base-Class ptr of the DataImpl belonging to the datatype

14.3.3.10 boost::shared_ptr<Data> MetNoFimex::createData (size_t *length*, boost::shared_array< float > *array*)

create a Data-pointer of type CDM_FLOAT

Parameters

size_t length of the data array

array the data array

Returns

Base-Class ptr of the DataImpl belonging to the datatype

14.3.3.11 boost::shared_ptr<Data> MetNoFimex::createData (size_t *length*, boost::shared_array< double > *array*)

create a Data-pointer of type CDM_DOUBLE

Parameters

size_t length of the data array

array the data array

Returns

Base-Class ptr of the DataImpl belonging to the datatype

14.3.3.12 boost::shared_ptr<Data> MetNoFimex::createData (CDMDataType *datatype*, size_t *length*, double *val* = 0)

create a Data-pointer of the datatype

Parameters

datatype

size_t length of the data array

val default value for data elements, 0 by default

Returns

Base-Class ptr of the DataImpl belonging to the datatype

Referenced by createData().

14.3.3.13 boost::shared_ptr<Data> MetNoFimex::createDataSlice (CDMDataType *datatype*, const Data & *data*, size_t *dataStartPos*, size_t *dataSize*)

create a one-dimensional dataslice from another Data object

Parameters

datatype of the return-data

data the data to read the values from, should be convertible data-format

dataStartPos the first element of data to fetch

dataSize the size of the data

14.3.3.14 std::string MetNoFimex::datatype2string (CDMDataType *type*)

14.3.3.15 void MetNoFimex::defaultLogLevel (Logger::LogLevel)

14.3.3.16 Logger::LogLevel MetNoFimex::defaultLogLevel ()

the defaultLogLevel can be used by the implemented logger to determine the minimum LogLevel. This value might be ignored/overwritten by a configuration within the implementation. It should be initialized in the main class.

14.3.3.17 MetNoFimex::DEPRECATED (std::string *attributesToProjString*const std::vector< CDMAttribute > &attrs)

convert attributes of a projection-variable to a projString

Deprecated

use Projection::create() with Projection::getProj4String() instead

Parameters

attrs attributes of the projection variable

Returns

proj4 string

14.3.3.18 MetNoFimex::DEPRECATED (std::vector< CDMAttribute > *projStringToAttributes*std::string projStr)

convert a proj4 string to a list of CDMAttributes usable for CF-1.0 projection variable

currently, projStrings of the form +proj=[stere] +lat_0=? +lon_0=? +lat_ts=?

Deprecated

use Projection::createByProj4() and Projection::getParameters()

14.3.3.19 template<typename InputIterator > std::pair<typename std::iterator_traits<InputIterator>::difference_type, typename std::iterator_traits<InputIterator>::difference_type> MetNoFimex::find_closest_distinct_elements (InputIterator *start*, InputIterator *end*, double *x*) [inline]

Find closest distinct elements in an unordered list. The order of elements is not defined.

Except for the case where all elements are equal, it is always ensured that the neighbors are distinct.

Parameters

start

end

Returns

pair of the positions of a and b, with a closer than b

References distance().

Referenced by find_closest_neighbor_distinct_elements().

14.3.3.20 template<typename InputIterator > std::pair<typename std::iterator_traits<InputIterator>::difference_type, typename std::iterator_traits<InputIterator>::difference_type> MetNoFimex::find_closest_neighbor_distinct_elements (InputIterator *start*, InputIterator *end*, double *x*) [inline]

Find closest distinct neighbor elements in an unordered list, with a <= x < b It might extrapolate if x is smaller than all elements (or x > all elements) and fall back to find_closest_distinct_elements()

Except for the case where all elements are equal, it is always ensured that the neighbors are distinct.

Parameters

start

end

Returns

pair of the positions of a and b, with a closer than b

References distance(), and find_closest_distinct_elements().

14.3.3.21 LoggerPtr MetNoFimex::getLogger (const std::string & *className*)

Retrieve a logger for Fimex. It will use loggers in the following order, skipping to the next one if the current one is not available: 1) log4cxx 2) no/dummy logger

14.3.3.22 boost::posix_time::ptime MetNoFimex::getUniqueForecastReferenceTime (boost::shared_ptr< CDMReader > *reader*)

Try to find the forecast reference time of the reader. This has currently only be implemented for CF-1.x.

Parameters

reader the CDMReader to check for the reference time

Returns

the reference time

Exceptions

CDMException if eithern no reference time has been found, or if more than 1 different reference times have been found

14.3.3.23 std::string MetNoFimex::getXmlContent (const xmlNodePtr *node*)

get all text-contents of the node or underlying nodes

Parameters

node the xmlNodePtr or xmlNodePtr as list

Returns

string with text-content, or ""

Exceptions

CDMException

14.3.3.24 std::string MetNoFimex::getXmlName (const xmlNodePtr *node*)

a memory-save form of xmlGetName

Returns

a string of the attribute, "" if attribute doesn't exist

14.3.3.25 std::string MetNoFimex::getXmlProp (const xmlNodePtr *node*, const std::string & *attrName*)

a memory-save form of xmlGetProp

Returns

a string of the attribute, "" if attribute doesn't exist

14.3.3.26 GridDefinition::Orientation MetNoFimex::gribGetGridOrientation (boost::shared_ptr< grib_handle > *gh*)

get the orientation of the data

Parameters

gh grib-handle

14.3.3.27 void MetNoFimex::handleUdUnitError (int *unitErrCode*, const std::string & *message* = "") throw (UnitException)

14.3.3.28 template<class InputIterator > std::string MetNoFimex::join (InputIterator *start*, InputIterator *end*, std::string *delim* = ",") [inline]

Join values from an iterator to a string, using delimiter as separator.

Parameters

start

end

delim separator, default to ","

14.3.3.29 template<class InputIterator > std::string MetNoFimex::joinPtr (InputIterator *start*, InputIterator *end*, std::string *delim* = ",") [inline]

Join values from an iterator of pointers to a string, using delimiter as separator.

Parameters

start

end

delim separator, default to ","

14.3.3.30 std::vector<boost::shared_ptr<const CoordinateSystem> > MetNoFimex::listCoordinateSystems (const CDM & *cdm*)

fetch all coordinate system from a MetNoFimex::CDM

Examples:

coordinateSystem.cpp.

14.3.3.31 template<typename T > boost::shared_array<const T> MetNoFimex::makeSharedArrayConst (const boost::shared_array< T > & *sa*) [inline]

convert a shared_array<T> to a shared_array<const T> (which will be automatically possilbe in boost::shared_array 1.47)

14.3.3.32 std::ostream& MetNoFimex::operator<< (std::ostream & *out*, const FimexTime & *fTime*)

minimum FimexTime

14.3.3.33 std::ostream& MetNoFimex::operator<< (std::ostream & *os*, const GribFileIndex & *gfm*)

outputstream for a GribFileIndex

14.3.3.34 std::ostream& MetNoFimex::operator<< (std::ostream & *os*, const GribFileMessage & *gfm*)

outputstream for a GribFileMessage

14.3.3.35 std::ostream& MetNoFimex::operator<< (std::ostream & *out*, const Projection & *proj*)

output-stream for projections, implemented using toString()

14.3.3.36 std::ostream& MetNoFimex::operator<< (std::ostream & *out*, const CoordinateSystem & *p*)

output operator

14.3.3.37 std::ostream& MetNoFimex::operator<< (std::ostream & *out*, CoordinateAxis::AxisType *t*)

14.3.3.38 std::ostream& MetNoFimex::operator<< (std::ostream & *out*, CoordinateAxis *ca*)

14.3.3.39 epoch_seconds MetNoFimex::posixTime2epochTime (const boost::posix_time::ptime & *time*)

convert a posixTime to seconds sinc 1970-01-01

Parameters

time time to convert

14.3.3.40 int MetNoFimex::round (double *num*)

Round a double to integer.

14.3.3.41 CDMDataType MetNoFimex::string2datatype (const std::string & *s*)

translate float/string/... to the appropriate CDMDataType

14.3.3.42 FimexTime MetNoFimex::string2FimexTime (const std::string & *str*) throw (CDMException)

14.3.3.43 std::string MetNoFimex::string2lowerCase (const std::string & *str*)

convert a string to lowercase

14.3.3.44 template<typename T > T MetNoFimex::string2type (std::string *s*) [inline]

14.3.3.45 std::vector<std::string> MetNoFimex::tokenize (const std::string & *str*, const std::string & *delimiters* = " ")

Tokenize a string by a delimiter. This function will automaticall remove empty strings at the beginning or anywhere inside the string.

This function has been derived from http://www.oopweb.com/CPP/Documents/CPPHOWTO/Volume/C++Progra

Parameters

str the string to tokenize

delimiters the delimiters between the tokens. That can be multiple delimiters, i.e. whitespace is " \t\n\r"

Returns

vector of tokens

Referenced by tokenizeDotted().

14.3.3.46 template<typename T > std::vector<T> MetNoFimex::tokenizeDotted (const std::string & *str*, const std::string & *delimiter* = ",") throw (CDMException) [inline]

convert a string with dots to a vector with type T

Parameters

str f.e. 3.5,4.5,...,17.5

delimiter optional delimiter, defaults to ,

References std::vector< _Tp, _Alloc >::begin(), std::vector< _Tp, _Alloc >::end(), std::vector< _Tp, _Alloc >::push_back(), std::vector< _Tp, _Alloc >::size(), tokenize(), trim(), and type2string().

14.3.3.47 std::string MetNoFimex::trim (const std::string & *str*)

Remove leading and trailing spaces.

Parameters

str string to trim

Referenced by tokenizeDotted().

14.3.3.48 template<typename T > std::string MetNoFimex::type2string (T *in*) [inline]

convert a type (i.e. int, float) to string representation

Referenced by tokenizeDotted().

14.3.3.49 template<> std::string MetNoFimex::type2string< double > (double *in*) [inline]

specialization for high prececision

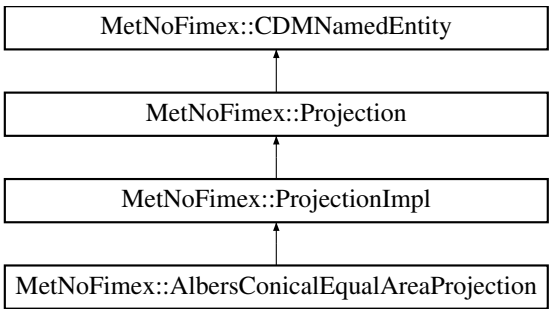
Chapter 15

Class Documentation

15.1 MetNoFimex::AlbersConicalEqualAreaProjection Class Reference

`#include <AlbersConicalEqualAreaProjection.h>`

Inheritance diagram for MetNoFimex::AlbersConicalEqualAreaProjection:



Public Member Functions

- AlbersConicalEqualAreaProjection ()
- virtual ~AlbersConicalEqualAreaProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- AlbersConicalEqualAreaProjection (**std::string** name)
- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.1.1 Constructor & Destructor Documentation

15.1.1.1 MetNoFimex::AlbersConicalEqualAreaProjection::AlbersConicalEqualAreaProjection ()

15.1.1.2 virtual MetNoFimex::AlbersConicalEqualAreaProjection::~AlbersConicalEqualAreaProjection () [inline, virtual]

15.1.1.3 MetNoFimex::AlbersConicalEqualAreaProjection::AlbersConicalEqualAreaProjection (std::string *name*) [inline, protected]

15.1.2 Member Function Documentation

15.1.2.1 static bool MetNoFimex::AlbersConicalEqualAreaProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

15.1.2.2 virtual std::ostream& MetNoFimex::AlbersConicalEqualAreaProjection::getProj4ProjectionPart (std::ostream &) const [protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.1.2.3 static std::vector<CDMAttribute> MetNoFimex::AlbersConicalEqualAreaProjection::parametersFromProj4 (const std::string & *proj4*) [static]

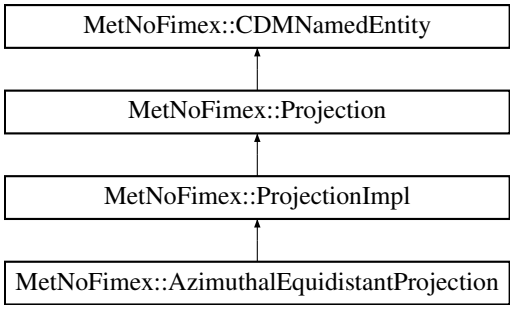
The documentation for this class was generated from the following file:

- include/fimex/coordSys/AlbersConicalEqualAreaProjection.h

15.2 MetNoFimex::AzimuthalEquidistantProjection Class Reference

`#include <AzimuthalEquidistantProjection.h>`

Inheritance diagram for MetNoFimex::AzimuthalEquidistantProjection:



Public Member Functions

- AzimuthalEquidistantProjection ()
- virtual ~AzimuthalEquidistantProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- AzimuthalEquidistantProjection (**std::string** name)
- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.2.1 Constructor & Destructor Documentation

15.2.1.1 MetNoFimex::AzimuthalEquidistantProjection::AzimuthalEquidistantProjection ()

15.2.1.2 virtual MetNoFimex::AzimuthalEquidistantProjection::~AzimuthalEquidistantProjection () [inline, virtual]

15.2.1.3 MetNoFimex::AzimuthalEquidistantProjection::AzimuthalEquidistantProjection (std::string *name*) [inline, protected]

15.2.2 Member Function Documentation

15.2.2.1 static bool MetNoFimex::AzimuthalEquidistantProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

15.2.2.2 virtual std::ostream& MetNoFimex::AzimuthalEquidistantProjection::getProj4ProjectionPart (std::ostream &) const [protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.2.2.3 static std::vector<CDMAttribute> MetNoFimex::AzimuthalEquidistantProjection::parametersFromProj4 (const std::string & *proj4*) [static]

The documentation for this class was generated from the following file:

- include/fimex/coordSys/AzimuthalEquidistantProjection.h

15.3 binary< N > Struct Template Reference

`#include <binaryConstants.h>`

Public Types

- enum { value = (N % 8) + (binary<N/8>::value << 1) }

15.3.1 Detailed Description

template<unsigned long long N> struct binary< N >

use binary<01001001>::value as constant, works with up to 10 bits use binary<01001001ULL>::value as constant, works with up to 22 bits

Warning

always start with leading 0, since all values have to be octals!!!

15.3.2 Member Enumeration Documentation

15.3.2.1 template<unsigned long long N> anonymous enum

Enumerator:

value

The documentation for this struct was generated from the following file:

- include/fimex/binaryConstants.h

15.4 binary< 0 > Struct Template Reference

`#include <binaryConstants.h>`

Public Types

- enum { value = 0 }

template<> struct binary< 0 >

15.4.1 Member Enumeration Documentation

15.4.1.1 anonymous enum

Enumerator:

value

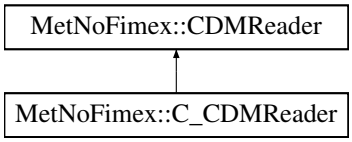
The documentation for this struct was generated from the following file:

- include/fimex/binaryConstants.h

15.5 MetNoFimex::C_CDMReader Class Reference

`#include <C_CDMReader.h>`

Inheritance diagram for MetNoFimex::C_CDMReader:



Public Member Functions

- C_CDMReader (boost::shared_ptr< CDMReader > dataReader)
- virtual ~C_CDMReader ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos)

  data-reading function to be called from the CDMWriter

- virtual void setDoubleCallbackFunction (const **std::string** &varName, doubleDatasliceCallbackPtr callback)

15.5.1 Detailed Description

This class should be used by people who want write an implementation of a CDMReader in C. They should set a callback-function to retrieve a variable with the getDataSlice functions.

15.5.2 Constructor & Destructor Documentation

15.5.2.1 MetNoFimex::C_CDMReader::C_CDMReader (boost::shared_ptr< CDMReader > *dataReader*)

15.5.2.2 virtual MetNoFimex::C_CDMReader::~C_CDMReader () `[virtual]`

15.5.3 Member Function Documentation

15.5.3.1 virtual boost::shared_ptr<Data> MetNoFimex::C_CDMReader::getDataSlice (const std::string & *varName*, size_t *unLimDimPos*) `[virtual]`

data-reading function to be called from the CDMWriter

This methods needs to be implemented by the CDMReader. It should provide the data for each variable, either by reading from disk, converting from another CDMReader or reading from an in-memory data-section.

This function should retrieve the whole data for a dataset without unlimited dimension if the unLimDimPos == 0.

Parameters

varName name of the variable to read

unLimDimPos (optional) if the variable contains a unlimited dimension (max one allowed) an slice of this position is returned

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

Implements MetNoFimex::CDMReader.

15.5.3.2 virtual void MetNoFimex::C_CDMReader::setDoubleCallbackFunction (const std::string & *varName*, doubleDatasliceCallbackPtr *callback*) [virtual]

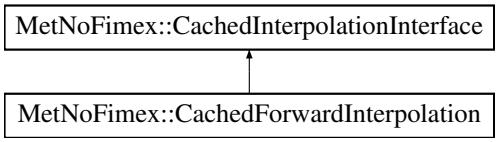
The documentation for this class was generated from the following file:

- include/fimex/C_CDMReader.h

15.6 MetNoFimex::CachedForwardInterpolation Class Reference

`#include <CachedForwardInterpolation.h>`

Inheritance diagram for MetNoFimex::CachedForwardInterpolation:



Public Member Functions

- CachedForwardInterpolation (int funcType, **std::vector**< double > pointsOnXAxis, **std::vector**< double > pointsOnYAxis, size_t inX, size_t inY, size_t outX, size_t outY)
- virtual ~CachedForwardInterpolation ()
- virtual boost::shared_array< float > interpolateValues (boost::shared_array< float > inData, size_t size, size_t &newSize) const
- virtual size_t getInX () const
- virtual size_t getInY () const

15.6.1 Constructor & Destructor Documentation

15.6.1.1 MetNoFimex::CachedForwardInterpolation::CachedForwardInterpolation (int *funcType*, std::vector< double > *pointsOnXAxis*, std::vector< double > *pointsOnYAxis*, size_t *inX*, size_t *inY*, size_t *outX*, size_t *outY*)

15.6.1.2 virtual MetNoFimex::CachedForwardInterpolation::~CachedForwardInterpolation () `[inline, virtual]`

15.6.2 Member Function Documentation

15.6.2.1 virtual size_t MetNoFimex::CachedForwardInterpolation::getInX () const `[inline, virtual]`

return x-size of input array

Implements MetNoFimex::CachedInterpolationInterface.

15.6.2.2 virtual size_t MetNoFimex::CachedForwardInterpolation::getInY () const `[inline, virtual]`

return y-size of input array

Implements MetNoFimex::CachedInterpolationInterface.

15.6.2.3 virtual boost::shared_array<float> MetNoFimex::CachedForwardInterpolation::interpolateValues (boost::shared_array< float > *inData*, size_t *size*, size_t & *newSize*) const [virtual]

Implements MetNoFimex::CachedInterpolationInterface.

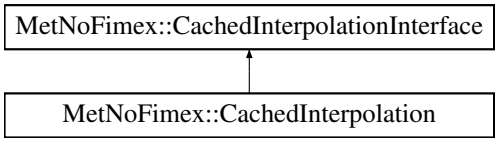
The documentation for this class was generated from the following file:

- include/fimex/CachedForwardInterpolation.h

15.7 MetNoFimex::CachedInterpolation Class Reference

`#include <CachedInterpolation.h>`

Inheritance diagram for MetNoFimex::CachedInterpolation:



Public Member Functions

- CachedInterpolation (int funcType, **std::vector**< double > pointsOnXAxis, **std::vector**< double > pointsOnYAxis, size_t inX, size_t inY, size_t outX, size_t outY)
- virtual ~CachedInterpolation ()
- virtual boost::shared_array< float > interpolateValues (boost::shared_array< float > inData, size_t size, size_t &newSize) const
- virtual size_t getInX () const
- virtual size_t getInY () const

15.7.1 Detailed Description

Container to cache projection details to speed up interpolation of lots of fields.

15.7.2 Constructor & Destructor Documentation

15.7.2.1 MetNoFimex::CachedInterpolation::CachedInterpolation (int *funcType*, std::vector< double > *pointsOnXAxis*, std::vector< double > *pointsOnYAxis*, size_t *inX*, size_t *inY*, size_t *outX*, size_t *outY*)

Parameters

funcType interpolation.h interpolation method

pointsOnXAxis projected values of the new projections coordinates expressed in the current x-coordinate (size = outX∗outY)

pointsOnYAxis projected values of the new projections coordinates expressed in the current y-coordinate (size = outX∗outY)

inX size of current X axis

inY size of current Y axis

outX size of new X axis

outY size of new Y axis

15.7.2.2 virtual MetNoFimex::CachedInterpolation::~CachedInterpolation () [inline, virtual]

15.7.3 Member Function Documentation

15.7.3.1 virtual size_t MetNoFimex::CachedInterpolation::getInX () const [inline, virtual]

return x-size of input array

Implements MetNoFimex::CachedInterpolationInterface.

15.7.3.2 virtual size_t MetNoFimex::CachedInterpolation::getInY () const [inline, virtual]

return y-size of input array

Implements MetNoFimex::CachedInterpolationInterface.

15.7.3.3 virtual boost::shared_array<float> MetNoFimex::CachedInterpolation::interpolateValues (boost::shared_array< float > *inData*, size_t *size*, size_t & *newSize*) const [virtual]

Actually interpolate the data. The data will be interpolated as floats internally.

Parameters

inData the input data

the size of the input data array

newSize return the size of the output-array

Implements MetNoFimex::CachedInterpolationInterface.

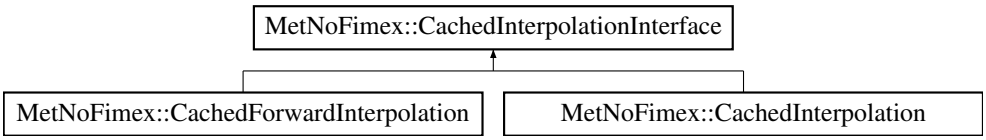
The documentation for this class was generated from the following file:

- include/fimex/CachedInterpolation.h

15.8 MetNoFimex::CachedInterpolationInterface Class Reference

`#include <CachedInterpolation.h>`

Inheritance diagram for MetNoFimex::CachedInterpolationInterface:



Public Member Functions

- virtual boost::shared_array< float > interpolateValues (boost::shared_array< float > inData, size_t size, size_t &newSize) const =0
- virtual size_t getInX () const =0
- virtual size_t getInY () const =0

15.8.1 Detailed Description

Interface for new cached spatial interpolation as used in MetNoFimex::CDMInterpolator

15.8.2 Member Function Documentation

15.8.2.1 virtual size_t MetNoFimex::CachedInterpolationInterface::getInX () const [pure virtual]

return x-size of input array

Implemented in MetNoFimex::CachedForwardInterpolation, and MetNoFimex::CachedInterpolation.

15.8.2.2 virtual size_t MetNoFimex::CachedInterpolationInterface::getInY () const [pure virtual]

return y-size of input array

Implemented in MetNoFimex::CachedForwardInterpolation, and MetNoFimex::CachedInterpolation.

15.8.2.3 virtual boost::shared_array<float> MetNoFimex::CachedInterpolationInterface::interpolateValues (boost::shared_array< float > *inData*, size_t *size*, size_t & *newSize*) const [pure virtual]

Implemented in MetNoFimex::CachedForwardInterpolation, and MetNoFimex::CachedInterpolation.

The documentation for this class was generated from the following file:

- include/fimex/CachedInterpolation.h

15.9 MetNoFimex::CachedVectorReprojection Class Reference

`#include <CachedVectorReprojection.h>`

Public Member Functions

- CachedVectorReprojection ()
- CachedVectorReprojection (int method, boost::shared_array< double > matrix, int ox, int oy)
- virtual ~CachedVectorReprojection ()
- void reprojectValues (boost::shared_array< float > &uValues, boost::shared_array< float > &vValues, size_t size) const throw (CDMException)
- size_t getXSize () const
- size_t getYSize () const

15.9.1 Constructor & Destructor Documentation

15.9.1.1 MetNoFimex::CachedVectorReprojection::CachedVectorReprojection () [inline]

15.9.1.2 MetNoFimex::CachedVectorReprojection::CachedVectorReprojection (int *method*, boost::shared_array< double > *matrix*, int *ox*, int *oy*) [inline]

15.9.1.3 virtual MetNoFimex::CachedVectorReprojection::~CachedVectorReprojection () [inline, virtual]

15.9.2 Member Function Documentation

15.9.2.1 size_t MetNoFimex::CachedVectorReprojection::getXSize () const [inline]

15.9.2.2 size_t MetNoFimex::CachedVectorReprojection::getYSize () const [inline]

15.9.2.3 void MetNoFimex::CachedVectorReprojection::reprojectValues (boost::shared_array< float > & *uValues*, boost::shared_array< float > & *vValues*, size_t *size*) const throw (CDMException)

reproject the vector values

Parameters

uValues the values in x-direction. These will be changed in-place.

vValues the values in y-direction. These will be changed in-place.

size the size of both arrays

The documentation for this class was generated from the following file:

- include/fimex/CachedVectorReprojection.h

15.10 MetNoFimex::CDM Class Reference

Data structure of the Common Data Model.

`#include "fimex/CDM.h"`

Public Types

- typedef **std::vector**< CDMAttribute > AttrVec
- typedef **std::map**< **std::string**, **AttrVec** > StrAttrVecMap
- typedef **std::vector**< CDMDimension > DimVec
- typedef **std::vector**< CDMVariable > VarVec

Public Member Functions

- CDM ()
- CDM (const CDM &rhs)
- virtual ~CDM ()
- CDM & operator= (const CDM &rhs)
- void addVariable (const CDMVariable &var) throw (CDMException)

  add variable to cdm

- CDMVariable & getVariable (const **std::string** &varName) throw (CDMException)

  get a reference of a variable

- const CDMVariable & getVariable (const **std::string** &varName) const throw (CDMException)

  get a reference of a variable

- bool hasVariable (const **std::string** &varName) const

  test if variable exists

- **std::vector**< **std::string** > findVariables (const **std::string** &attrName, const **std::string** &attrValueRegExp) const

  search for variable with certain attribute-value

- **std::vector**< **std::string** > findVariables (const **std::map**< **std::string**, **std::string** > &findAttributes, const **std::vector**< **std::string** > &findDimensions) const

  search for variable with attribute-values and dimensions

- bool renameVariable (const **std::string** &oldName, const **std::string** &newName)

  rename a variable

- bool checkVariableAttribute (const **std::string** &varName, const **std::string** &attribute, const boost::regex &attrValue) const
- void removeVariable (const **std::string** &variableName)

  remove a variable and corresponding attributes

- void addDimension (const CDMDimension &dim) throw (CDMException)

  add a dimension to cdm

- bool hasDimension (const **std::string** &dimName) const
- CDMDimension & getDimension (const **std::string** &dimName) throw (CDMException)

  get a reference to a dimension

- const CDMDimension & getDimension (const **std::string** &dimName) const throw (CDMException)
- bool testDimensionInUse (const **std::string** &name) const

  test if a dimension is actively in use

- bool renameDimension (const **std::string** &oldName, const **std::string** &newName) throw (CDMException)

  rename a dimension

- bool removeDimension (const **std::string** &name) throw (CDMException)

  remove a dimension

- const CDMDimension * getUnlimitedDim () const

  retrieve the unlimited dimension

- bool hasUnlimitedDim (const CDMVariable &var) const

  test if a variable contains the unlimited dim

- void addAttribute (const **std::string** &varName, const CDMAttribute &attr) throw (CDMException)
- void addOrReplaceAttribute (const **std::string** &varName, const CDMAttribute &attr) throw (CDMException)
- void removeAttribute (const **std::string** &varName, const **std::string** &attrName)
- void toXMLStream (**std::ostream** &os) const

  print a xml representation to the stream

- const **DimVec** & getDimensions () const

  get the dimension

- const **VarVec** & getVariables () const

  get the variables

- const **StrAttrVecMap** & getAttributes () const

  get the attributes

- **std::vector**< CDMAttribute > getAttributes (const **std::string** &varName) const

  get the attributes of an variable

- CDMAttribute & getAttribute (const **std::string** &varName, const **std::string** &attrName) throw (CDMException)

  get an attribute

- const CDMAttribute & getAttribute (const **std::string** &varName, const **std::string** &attrName) const throw (CDMException)

  get a const. attribute

- bool getAttribute (const **std::string** &varName, const **std::string** &attrName, CDMAttribute &retAttribute) const

get an attribute without throwing an error

- double getFillValue (const **std::string** &varName) const
- double getValidMin (const **std::string** &varName) const
- double getValidMax (const **std::string** &varName) const
- **std::string** getUnits (const **std::string** &varName) const
- void generateProjectionCoordinates (const **std::string** &projectionVariable, const **std::string** &xDim, const **std::string** &yDim, const **std::string** &lonDim, const **std::string** &latDim) throw (CDMException)

  generate the projection coordinates (usually named "lat lon")

- DEPRECATED (bool getProjectionAndAxesUnits(**std::string** &projectionName, **std::string** &xAxis, **std::string** &yAxis, **std::string** &xAxisUnits, **std::string** &yAxisUnits) const throw(CDMException))

  extract the names of the projection-variable and the corresponding projection-axes

- DEPRECATED (**AttrVec** getProjection(**std::string** varName) const)

  get the projection attributes (as of CF-1.0) of a variable

- boost::shared_ptr< const Projection > getProjectionOf (**std::string** varName) const

  get the projection of a variable

- **std::string** getHorizontalXAxis (**std::string** varName) const

  get the x-(lon) axis of the variable

- **std::string** getHorizontalYAxis (**std::string** varName) const

  get the y-(lat) axis of the variable

- bool getLatitudeLongitude (**std::string** varName, **std::string** &latitude, **std::string** &longitude) const

  detect the latitude and longitude coordinates of the variable

- **std::string** getTimeAxis (**std::string** varName) const

  get the time axis of the variable

- **std::string** getVerticalAxis (**std::string** varName) const

  get the vertical axis of the variable

Static Public Member Functions

- static const **std::string** & globalAttributeNS ()

  the namespace for global attributes

15.10.1 Detailed Description

Data structure of the Common Data Model. This class implements the data-structure of the Common Data Model version 1 http://www.unidata.ucar.edu/software/netcdf-java/CDM.html

Examples:

coordinateSystem.cpp.

15.10.2 Member Typedef Documentation

15.10.2.1 typedef std::vector<CDMAttribute> MetNoFimex::CDM::AttrVec

15.10.2.2 typedef std::vector<CDMDimension> MetNoFimex::CDM::DimVec

15.10.2.3 typedef std::map<std::string, AttrVec> MetNoFimex::CDM::StrAttrVecMap

15.10.2.4 typedef std::vector<CDMVariable> MetNoFimex::CDM::VarVec

15.10.3 Constructor & Destructor Documentation

15.10.3.1 MetNoFimex::CDM::CDM ()

15.10.3.2 MetNoFimex::CDM::CDM (const CDM & *rhs*)

15.10.3.3 virtual MetNoFimex::CDM::~CDM () `[virtual]`

15.10.4 Member Function Documentation

15.10.4.1 void MetNoFimex::CDM::addAttribute (const std::string & *varName*, const CDMAttribute & *attr*) throw (CDMException)

add an attribute to cdm

Parameters

varName name of the variablt the attribute belongs to

attr the CDMAttribute

Exceptions

CDMException if varName doesn't exist, or attr.getName() already exists

15.10.4.2 void MetNoFimex::CDM::addDimension (const CDMDimension & *dim*) throw (CDMException)

add a dimension to cdm

Parameters

dim the dimension

Exceptions

CDMException if dim-name already exists

15.10.4.3 void MetNoFimex::CDM::addOrReplaceAttribute (const std::string & *varName*, const CDMAttribute & *attr*) throw (CDMException)

add or replace an attribute of the cdm

Parameters

varName name of variable the attribute belongs to

attr the CDMAttribute

Exceptions

CDMException if vaName doesn't exist

15.10.4.4 void MetNoFimex::CDM::addVariable (const CDMVariable & *var*) throw (CDMException)

add variable to cdm

Parameters

var the variable to add

Exceptions

CDMException if var.varName() already exists

15.10.4.5 bool MetNoFimex::CDM::checkVariableAttribute (const std::string & *varName*, const std::string & *attribute*, const boost::regex & *attrValue*) const

check if a variable contains a attributes with a matching string-value

Parameters

varName variable

attribute the attribute name

attrValue the regexp the string-value of the attribute will match against

15.10.4.6 MetNoFimex::CDM::DEPRECATED (AttrVec getProjection(std::string varName) *const*)

get the projection attributes (as of CF-1.0) of a variable

Parameters

varName name of variable

Returns

vector of attributes of the projection, an empty vector if no projection found

Deprecated

use the getProjectionOf() method

15.10.4.7 MetNoFimex::CDM::DEPRECATED (bool *getProjectionAndAxesUnits*std::string &projectionName, std::string &xAxis, std::string &yAxis, std::string &xAxisUnits, std::string &yAxisUnits) const throw(CDMException)

extract the names of the projection-variable and the corresponding projection-axes

Parameters

projectionName output of the projection variables name

xAxis output of the spatial x axis

yAxis output of the spation y axis

xAxisUnit output of unit for x axis

yAxisUnit output of unit for y axis

Returns

true if unique result, false (and print warning) if results are not unique

Exceptions

CDMException if no projection with corresponding axes can be found

15.10.4.8 std::vector<std::string> MetNoFimex::CDM::findVariables (const std::map< std::string, std::string > & *findAttributes*, const std::vector< std::string > & *findDimensions*) const

search for variable with attribute-values and dimensions

And AND search for attributes and dimensions.

Parameters

findAttributes map with (attribute => string-value regExp) pairs

findDimensions vector with dimensions contained in variable

Returns

variable names of the variable with attributes matching the request and containing all dimensions

15.10.4.9 std::vector<std::string> MetNoFimex::CDM::findVariables (const std::string & *attrName*, const std::string & *attrValueRegExp*) const

search for variable with certain attribute-value

Parameters

attrName name of the attribute

attrValueRegExp regular expression the 'string'-value needs to match

Returns

variable names of the variable with attributes matching

15.10.4.10 void MetNoFimex::CDM::generateProjectionCoordinates (const std::string & *projectionVariable*, const std::string & *xDim*, const std::string & *yDim*, const std::string & *lonDim*, const std::string & *latDim*) throw (CDMException)

generate the projection coordinates (usually named "lat lon")

Parameters

projectionVariable the variable containing the projection information

xDim the x dimension (the corresponding variable needs to contain data and units)

yDim the y dimension (the corresponding variable needs to contain data and units)

lonDim name of the longitude variable

latDim name of the latitude variable

Exceptions

CDMException if any information is missing

15.10.4.11 bool MetNoFimex::CDM::getAttribute (const std::string & *varName*, const std::string & *attrName*, CDMAttribute & *retAttribute*) const

get an attribute without throwing an error

This method will search for an attribute in the cdm. It will return true on success and return the attribute.

Parameters

varName name of variable

attrName name of attribute

retAttribute returns the attribute if found

Returns

true when attribute has been found and set

15.10.4.12 const CDMAttribute& MetNoFimex::CDM::getAttribute (const std::string & *varName*, const std::string & *attrName*) const throw (CDMException)

get a const. attribute

Parameters

varName name of variable

attrName name of attribute

Exceptions

CDMException if varName attrName combination doesn't exists

15.10.4.13 CDMAttribute& MetNoFimex::CDM::getAttribute (const std::string & *varName*, const std::string & *attrName*) throw (CDMException)

get an attribute

Parameters

varName name of variable

attrName name of attribute

Exceptions

CDMException if varName attrName combination doesn't exists

15.10.4.14 std::vector<CDMAttribute> MetNoFimex::CDM::getAttributes (const std::string & *varName*) const

get the attributes of an variable

Parameters

varName name of variable

15.10.4.15 const StrAttrVecMap& MetNoFimex::CDM::getAttributes () const

get the attributes

Returns

map of type <variableName <attributeName, attribute>>

15.10.4.16 const CDMDimension& MetNoFimex::CDM::getDimension (const std::string & *dimName*) const throw (CDMException)

15.10.4.17 CDMDimension& MetNoFimex::CDM::getDimension (const std::string & *dimName*) throw (CDMException)

get a reference to a dimension

Parameters

dimName name of the dimension

Exceptions

CDMException if dimension doesn't exist

15.10.4.18 const DimVec& MetNoFimex::CDM::getDimensions () const

get the dimension

15.10.4.19 double MetNoFimex::CDM::getFillValue (const std::string & *varName*) const

get the fill value of an variable (_FillValue attribute)

Returns

value of _FillValue attribute, or MIFI_UNDEFINED_D

15.10.4.20 std::string MetNoFimex::CDM::getHorizontalXAxis (std::string *varName*) const

get the x-(lon) axis of the variable

This is the same as using the CoordinateSystem::getGeoXAxis().

Parameters

varName name of variable

Returns

name of x-axis dimension (or "" if not defined)

15.10.4.21 std::string MetNoFimex::CDM::getHorizontalYAxis (std::string *varName*) const

get the y-(lat) axis of the variable

This is the same as using the CoordinateSystem::getGeoYAxis().

Parameters

varName name of variable

Returns

name of y-axis dimension (or "" if not defined)

15.10.4.22 bool MetNoFimex::CDM::getLatitudeLongitude (std::string *varName*, std::string & *latitude*, std::string & *longitude*) const

detect the latitude and longitude coordinates of the variable

This is the same as using the CoordinateSystem::findAxisOfType() with CoordinateAxis::Lon and CoordinateAxis::Lat.

Parameters

varName name of variable

latitude return value of the latitude

longitude return value of the longitude

Returns

true if latitude and longitude have been found

15.10.4.23 boost::shared_ptr<const Projection> MetNoFimex::CDM::getProjectionOf (std::string *varName*) const

get the projection of a variable

This is the same as using the CoordinateSystem::getProjection().

Parameters

varName name of variable

Returns

projection

15.10.4.24 std::string MetNoFimex::CDM::getTimeAxis (std::string *varName*) const

get the time axis of the variable

This is the same as using the CoordinateSystem::getTimeAxis().

Parameters

varName name of variable

Returns

name of time dimension (or "" if not defined)

15.10.4.25 std::string MetNoFimex::CDM::getUnits (const std::string & *varName*) const

get the value of the "units" attribute

Returns

unitsString or ""

15.10.4.26 const CDMDimension* MetNoFimex::CDM::getUnlimitedDim () const

retrieve the unlimited dimension

Returns

unLimDim pointer with the unlimited dimension, the pointer will be deleted with the CDM

15.10.4.27 double MetNoFimex::CDM::getValidMax (const std::string & *varName*) const

get the valid maximum value of an variable

Returns

value of valid_max or valid_range attribute, or MIFI_UNDEFINED_D

15.10.4.28 double MetNoFimex::CDM::getValidMin (const std::string & *varName*) const

get the valid minimum value of an variable

Returns

value of valid_min or valid_range attribute, or MIFI_UNDEFINED_D

15.10.4.29 const CDMVariable& MetNoFimex::CDM::getVariable (const std::string & *varName*) const throw (CDMException)

get a reference of a variable

this is a constant version of CDMVariable::getVariable}

Parameters

varName name of the variable

Exceptions

CDMException if varName doesn't exist

15.10.4.30 CDMVariable& MetNoFimex::CDM::getVariable (const std::string & *varName*) throw (CDMException)

get a reference of a variable

Parameters

varName name of the variable

Exceptions

CDMException if varName doesn't exist

15.10.4.31 const VarVec& MetNoFimex::CDM::getVariables () const

get the variables

15.10.4.32 std::string MetNoFimex::CDM::getVerticalAxis (std::string *varName*) const

get the vertical axis of the variable

This is the same as using the CoordinateSystem::getGeoZAxis().

Parameters

varName name of variable

Returns

name of vertical dimension (or "" if not defined)

15.10.4.33 static const std::string& MetNoFimex::CDM::globalAttributeNS () [inline, static]

the namespace for global attributes

15.10.4.34 bool MetNoFimex::CDM::hasDimension (const std::string & *dimName*) const

check if the dimension exists

Parameters

dimName name of the dimension

15.10.4.35 bool MetNoFimex::CDM::hasUnlimitedDim (const CDMVariable & *var*) const

test if a variable contains the unlimited dim

Returns

true/false

15.10.4.36 bool MetNoFimex::CDM::hasVariable (const std::string & *varName*) const

test if variable exists

Parameters

varName name of variable

15.10.4.37 CDM& MetNoFimex::CDM::operator= (const CDM & *rhs*)

15.10.4.38 void MetNoFimex::CDM::removeAttribute (const std::string & *varName*, const std::string & *attrName*)

remove an attribute from the cdm

Parameters

varName name of variable the attribute belongs to

attr the CDMAttribute

15.10.4.39 bool MetNoFimex::CDM::removeDimension (const std::string & *name*) throw (CDMException)

remove a dimension

Remove a dimension, if it is not in use by a variable.

Returns

true if dimension existed, false otherwise

Exceptions

CDMException if dimension in us in a variable

15.10.4.40 void MetNoFimex::CDM::removeVariable (const std::string & *variableName*)

remove a variable and corresponding attributes

Parameters

variableName the variable to remove

15.10.4.41 bool MetNoFimex::CDM::renameDimension (const std::string & *oldName*, const std::string & *newName*) throw (CDMException)

rename a dimension

Rename a dimension.

Returns

false if the original name does not exist.

Exceptions

CDMException if newName already in use in a variable but for a different dimension

15.10.4.42 bool MetNoFimex::CDM::renameVariable (const std::string & *oldName*, const std::string & *newName*)

rename a variable

Parameters

oldName the old name of the variable

newName the new name of the variable

Returns

1 on success (oldName exists), 0 on failure

Warning

this will not change the spatialVectorCounterPart of all other variables

15.10.4.43 bool MetNoFimex::CDM::testDimensionInUse (const std::string & *name*) const

test if a dimension is actively in use

Parameters

name dimensionName

15.10.4.44 void MetNoFimex::CDM::toXMLStream (std::ostream & *os*) const

print a xml representation to the stream

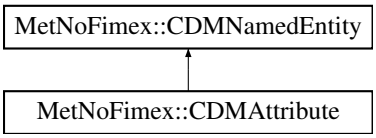
The documentation for this class was generated from the following file:

- include/fimex/CDM.h

15.11 MetNoFimex::CDMAttribute Class Reference

`#include <CDMAttribute.h>`

Inheritance diagram for MetNoFimex::CDMAttribute:



Public Member Functions

- CDMAttribute ()
- CDMAttribute (**std::string** name, **std::string** value)

  create a string attribute

- CDMAttribute (**std::string** name, char value)

  create a char attribute with a char array of length 1

- CDMAttribute (**std::string** name, int value)

  create a int attribute with a int array of length 1

- CDMAttribute (**std::string** name, short value)

  create a short attribute with a short array of length 1

- CDMAttribute (**std::string** name, float value)

  create a float attribute with a float array of length 1

- CDMAttribute (**std::string** name, double value)

  create a double attribute with a double array of length 1

- CDMAttribute (**std::string** name, CDMDataType datatype, boost::shared_ptr< Data > data)

  create a attribute with the low level information

- CDMAttribute (const **std::string** &name, const **std::string** &datatype, const **std::string** &value) throw (CDMException)

  create a attribute from a string representation

- CDMAttribute (const **std::string** &name, CDMDataType datatype, const **std::vector**< **std::string** > &values) throw (CDMException)

  create a attribute with a vector of values in string representation

- virtual ~CDMAttribute ()
- const **std::string** & getName () const

  retrieve the name of the attribute

- void setName (**std::string** newName)

  set the name of the attribute

- const **std::string** getStringValue () const

  retrieve the stringified value of the attribute

- const boost::shared_ptr< Data > getData () const

  retrieve the data-pointer of the attribute

- void setData (boost::shared_ptr< Data > data)

  set the data for this attribute

- const CDMDataType getDataType () const

  retrieve the datatype of the attribute

- void toXMLStream (**std::ostream** &out, const **std::string** &indent="") const

15.11.1 Constructor & Destructor Documentation

15.11.1.1 MetNoFimex::CDMAttribute::CDMAttribute ()

15.11.1.2 MetNoFimex::CDMAttribute::CDMAttribute (std::string *name*, std::string *value*) [explicit]

create a string attribute

15.11.1.3 MetNoFimex::CDMAttribute::CDMAttribute (std::string *name*, char *value*) [explicit]

create a char attribute with a char array of length 1

15.11.1.4 MetNoFimex::CDMAttribute::CDMAttribute (std::string *name*, int *value*) [explicit]

create a int attribute with a int array of length 1

15.11.1.5 MetNoFimex::CDMAttribute::CDMAttribute (std::string *name*, short *value*) [explicit]

create a short attribute with a short array of length 1

15.11.1.6 MetNoFimex::CDMAttribute::CDMAttribute (std::string *name*, float *value*) [explicit]

create a float attribute with a float array of length 1

15.11.1.7 MetNoFimex::CDMAttribute::CDMAttribute (std::string *name*, double *value*) [explicit]

create a double attribute with a double array of length 1

15.11.1.8 MetNoFimex::CDMAttribute::CDMAttribute (std::string *name*, CDMDataType *datatype*, boost::shared_ptr< Data > *data*) [explicit]

create a attribute with the low level information

15.11.1.9 MetNoFimex::CDMAttribute::CDMAttribute (const std::string & *name*, const std::string & *datatype*, const std::string & *value*) throw (CDMException) [explicit]

create a attribute from a string representation

15.11.1.10 MetNoFimex::CDMAttribute::CDMAttribute (const std::string & *name*, CDMDataType *datatype*, const std::vector< std::string > & *values*) throw (CDMException) [explicit]

create a attribute with a vector of values in string representation

15.11.1.11 virtual MetNoFimex::CDMAttribute::~CDMAttribute () [virtual]

15.11.2 Member Function Documentation

15.11.2.1 const boost::shared_ptr<Data> MetNoFimex::CDMAttribute::getData () const [inline]

retrieve the data-pointer of the attribute

15.11.2.2 const CDMDataType MetNoFimex::CDMAttribute::getDataType () const [inline]

retrieve the datatype of the attribute

15.11.2.3 const std::string& MetNoFimex::CDMAttribute::getName () const [inline, virtual]

retrieve the name of the attribute

Implements MetNoFimex::CDMNamedEntity.

15.11.2.4 const std::string MetNoFimex::CDMAttribute::getStringValue () const

retrieve the stringified value of the attribute

15.11.2.5 void MetNoFimex::CDMAttribute::setData (boost::shared_ptr< Data > *data*) [inline]

set the data for this attribute

15.11.2.6 void MetNoFimex::CDMAttribute::setName (std::string *newName*) [inline]

set the name of the attribute

15.11.2.7 void MetNoFimex::CDMAttribute::toXMLStream (std::ostream & *out*, const std::string & *indent* = "") const

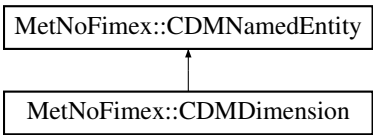
The documentation for this class was generated from the following file:

- include/fimex/CDMAttribute.h

15.12 MetNoFimex::CDMDimension Class Reference

`#include <CDMDimension.h>`

Inheritance diagram for MetNoFimex::CDMDimension:



Public Member Functions

- CDMDimension ()
- CDMDimension (**std::string** name, long length)
- virtual ~CDMDimension ()
- const **std::string** & getName () const
- void setName (**std::string** newName)
- size_t getLength () const
- void setLength (size_t length)
- void setUnlimited (int unlimited)
- int isUnlimited () const
- void toXMLStream (**std::ostream** &out) const

  print xml representation to stream

15.12.1 Constructor & Destructor Documentation

15.12.1.1 MetNoFimex::CDMDimension::CDMDimension ()

15.12.1.2 MetNoFimex::CDMDimension::CDMDimension (std::string *name*, long *length*)

15.12.1.3 virtual MetNoFimex::CDMDimension::~CDMDimension () `[virtual]`

15.12.2 Member Function Documentation

15.12.2.1 size_t MetNoFimex::CDMDimension::getLength () const `[inline]`

15.12.2.2 const std::string& MetNoFimex::CDMDimension::getName () const `[inline, virtual]`

Implements MetNoFimex::CDMNamedEntity.

15.12.2.3 int MetNoFimex::CDMDimension::isUnlimited () const [inline]

15.12.2.4 void MetNoFimex::CDMDimension::setLength (size_t *length*) [inline]

15.12.2.5 void MetNoFimex::CDMDimension::setName (std::string *newName*) [inline]

15.12.2.6 void MetNoFimex::CDMDimension::setUnlimited (int *unlimited*) [inline]

15.12.2.7 void MetNoFimex::CDMDimension::toXMLStream (std::ostream & *out*) const

print xml representation to stream

Parameters

out stream to write xml to

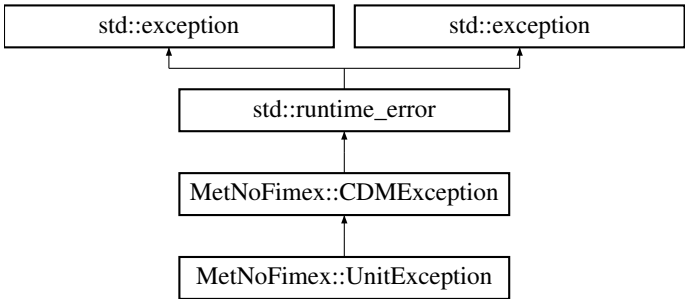
The documentation for this class was generated from the following file:

- include/fimex/CDMDimension.h

15.13 MetNoFimex::CDMException Class Reference

`#include <CDMException.h>`

Inheritance diagram for MetNoFimex::CDMException:



Public Member Functions

- CDMException ()
- CDMException (**std::string** msg)

15.13.1 Constructor & Destructor Documentation

15.13.1.1 MetNoFimex::CDMException::CDMException () [inline]

15.13.1.2 MetNoFimex::CDMException::CDMException (std::string *msg*) [inline, explicit]

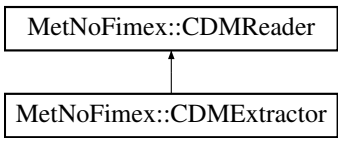
The documentation for this class was generated from the following file:

- include/fimex/CDMException.h

15.14 MetNoFimex::CDMExtractor Class Reference

`#include <CDMExtractor.h>`

Inheritance diagram for MetNoFimex::CDMExtractor:



Public Member Functions

- CDMExtractor (boost::shared_ptr< CDMReader > dataReader)
- virtual ~CDMExtractor ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos=0)

  data-reading function to be called from the CDMWriter

- virtual void removeVariable (**std::string** variable)

  Remove a variable from the CDM.

- virtual void selectVariables (**std::set**< **std::string** > variables)

  select only a set of variables

- virtual void reduceDimension (**std::string** dimName, size_t start, size_t length)

  Reduce a dimension of the file.

- virtual void reduceDimensionStartEnd (**std::string** dimName, size_t start=0, long end=0)

  Reduce a dimension of the file.

- virtual void reduceAxes (const **std::vector**< CoordinateAxis::AxisType > &types, const **std::string** &aUnits, double startVal, double endVal)

  reduce the axes of a file with an explicit unit

- virtual void reduceTime (const FimexTime &startTime, const FimexTime &endTime)

  reduce the time explicitly by a timestamp

- virtual void reduceVerticalAxis (const **std::string** &units, double startVal, double endVal)

  reduce a vertical axis by value

- virtual void reduceLatLonBoundingBox (double south, double north, double west, double east)

  reduce the horizontal layer to the latitude-longitude bounding box

- virtual void changeDataType (**std::string** variable, CDMDataType datatype)

  change the datatype of the variable

15.14.1 Constructor & Destructor Documentation

15.14.1.1 MetNoFimex::CDMExtractor::CDMExtractor (boost::shared_ptr< CDMReader > *dataReader*)

15.14.1.2 virtual MetNoFimex::CDMExtractor::~CDMExtractor () [virtual]

15.14.2 Member Function Documentation

15.14.2.1 virtual void MetNoFimex::CDMExtractor::changeDataType (std::string *variable*, CDMDataType *datatype*) [virtual]

change the datatype of the variable

a change of the variable will also change the datatype of the _FillValue attribute

Parameters

variable name of the variable

datatype new datatype

Exceptions

CDMException if variable doesn't exist or conversion to datatype is not supported

15.14.2.2 virtual boost::shared_ptr<Data> MetNoFimex::CDMExtractor::getDataSlice (const std::string & *varName*, size_t *unLimDimPos* = 0) [virtual]

data-reading function to be called from the CDMWriter

This methods needs to be implemented by the CDMReader. It should provide the data for each variable, either by reading from disk, converting from another CDMReader or reading from an in-memory data-section.

This function should retrieve the whole data for a dataset without unlimited dimension if the unLimDimPos == 0.

Parameters

varName name of the variable to read

unLimDimPos (optional) if the variable contains a unlimited dimension (max one allowed) an slice of this position is returned

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

Implements MetNoFimex::CDMReader.

15.14.2.3 virtual void MetNoFimex::CDMExtractor::reduceAxes (const std::vector< CoordinateAxis::AxisType > & *types*, const std::string & *aUnits*, double *startVal*, double *endVal*) [virtual]

reduce the axes of a file with an explicit unit

In contrast to reduceDimension, this method allows the usage of absolute values, not positions on the dimension. It will try to detect the reduction of dimensions as needed.

Warning

reduceAxes requires the times to be monotonic
reduceAxes requires the file to come with a known convention, e.g. CF, see listCoordinateSystems()
reduceAxes is not able to reduce multi-dimensional axes-dimensions, e.g. time(time, station), yet

15.14.2.4 virtual void MetNoFimex::CDMExtractor::reduceDimension (std::string *dimName*, size_t *start*, size_t *length*) [virtual]

Reduce a dimension of the file.

Parameters

name dimension to change
start start-position corresponding to the original dimension
size size of the new dimension

Exceptions

CDMException if dimension doesn't exist or start+size outside range of the original dimension

15.14.2.5 virtual void MetNoFimex::CDMExtractor::reduceDimensionStartEnd (std::string *dimName*, size_t *start* = 0, long *end* = 0) [virtual]

Reduce a dimension of the file.

Parameters

name dimension to change
start start-position corresponding to the original dimension, defaults to 0
end end-position of dimension, 0 means full size, negative values start from end

Exceptions

CDMException if dimension doesn't exist or start+size outside range of the original dimension

15.14.2.6 virtual void MetNoFimex::CDMExtractor::reduceLatLonBoundingBox (double *south*, double *north*, double *west*, double *east*) [virtual]

reduce the horizontal layer to the latitude-longitude bounding box

This method will try to reduce the horizontal layer to the given latitude/longitude bounding box. It requires the original data to have a simple geospatial gridded CoordinateSystem, i.e. CoordinateSystem::isSimpleSpatialGridded() and a projection mapping to lat/lon

Parameters

south southernmost border in dec. degree, -90 < south < north < 90
north northernmost border in dec. degree, -90 < south < north < 90
west westernmost border in dec. degree, -180 < west < east < 180
east easternmost border in dec. degree, -180 < west < east < 180

15.14.2.7 virtual void MetNoFimex::CDMExtractor::reduceTime (const FimexTime & *startTime*, const FimexTime & *endTime*) [virtual]

reduce the time explicitly by a timestamp

In contrast to reduceDimension, this method allows the usage of absolute times. It will try to detect the reduction of dimensions as needed

This is implemented using reduceAxes() and the TimeAxis type.

Warning

see warnings in reduceAxes()

15.14.2.8 virtual void MetNoFimex::CDMExtractor::reduceVerticalAxis (const std::string & *units*, double *startVal*, double *endVal*) [virtual]

reduce a vertical axis by value

In contrast to reduceDimension, this method allows the usage of vertical axes values having a compatible unit to units. It will try to detect the reduction of dimensions as needed.

Parameters

units the units of the start and end value. Only vertical axes with compatible units will be reduced.

startVal the lower value of the axis (included)

endVal the upper value of the axis (included)

This is implemented using reduceAxes() and the axis types: pressure, height, geoZ.

Warning

see warnings in reduceAxes()

15.14.2.9 virtual void MetNoFimex::CDMExtractor::removeVariable (std::string *variable*) [virtual]

Remove a variable from the CDM.

Parameters

varName name of the variable

Warning

ignores removal of non-existing variable

15.14.2.10 virtual void MetNoFimex::CDMExtractor::selectVariables (std::set< std::string > *variables*) [virtual]

select only a set of variables

This function will remove all variables except the ones selected plus eventually some auxiliary variables needed by the selected variables (not decided yet)

Parameters

variables list of variables-names

Warning

ignores selection of non-existing variable

The documentation for this class was generated from the following file:

- include/fimex/CDMExtractor.h

15.15 MetNoFimex::CDMFileReaderFactory Class Reference

`#include <CDMFileReaderFactory.h>`

Static Public Member Functions

- static int detectFileType (const **std::string** &fileName)

  detect the filetype of a input-file

- static boost::shared_ptr< CDMReader > create (int fileType, const **std::string** &fileName, const **std::string** &configFile="", const **std::vector**< **std::string** > &args=**std::vector**< **std::string** >())

  Factory for CDMReader of input-files.

- static boost::shared_ptr< CDMReader > create (const **std::string** &fileType, const **std::string** &fileName, const **std::string** &configFile="", const **std::vector**< **std::string** > &args=**std::vector**< **std::string** >())

  same as the other create(), but with a fileType string

- static boost::shared_ptr< CDMReader > create (int fileType, const **std::string** &fileName, const XMLInput &configXML, const **std::vector**< **std::string** > &args=**std::vector**< **std::string** >())

  Factory for CDMReader of input-files.

- static boost::shared_ptr< CDMReader > create (const **std::string** &fileType, const **std::string** &fileName, const XMLInput &configXML, const **std::vector**< **std::string** > &args=**std::vector**< **std::string** >())

  same as the other create(), but with a fileType string

15.15.1 Detailed Description

helper class to simplify file-reader detection and creation

15.15.2 Member Function Documentation

15.15.2.1 static boost::shared_ptr<CDMReader> MetNoFimex::CDMFileReaderFactory::create (const std::string & *fileType*, const std::string & *fileName*, const XMLInput & *configXML*, const std::vector< std::string > & *args* = std::vector< std::string >()) [static]

same as the other create(), but with a fileType string

15.15.2.2 static boost::shared_ptr<CDMReader> MetNoFimex::CDMFileReaderFactory::create (int *fileType*, const std::string & *fileName*, const XMLInput & *configXML*, const std::vector< std::string > & *args* = std::vector< std::string >()) [static]

Factory for CDMReader of input-files.

The function create reader tries to create a reader by filetype MIFI_FILETYPE_*. The optional arguments are defined by the different readers. Use default objects (empty string, empty vector) if arguments are not desired.

Parameters

fileNype,one of MIFI_FILETYPE_*, possibly read by detectFileType()

fileName,name of input type

configXML config source

options optional options for the CDMReader

Returns

pointer to CDMReader

Exceptions

CDMException if type not compiled in, or creation fails

Deprecated

use create(int fileType, const **std::string**& fileName, const XMLInput& configXML, const std::vector<std::string>& args = std::vector<std::string>())

15.15.2.3 static boost::shared_ptr<CDMReader> MetNoFimex::CDMFileReaderFactory::create (const std::string & *fileType*, const std::string & *fileName*, const std::string & *configFile* = "", const std::vector< std::string > & *args* = std::vector< std::string >()) [static]

same as the other create(), but with a fileType string

Deprecated

use create(const **std::string**& fileType, const **std::string**& fileName, const XMLInput& configXML, const std::vector<std::string>& args = std::vector<std::string>())

15.15.2.4 static boost::shared_ptr<CDMReader> MetNoFimex::CDMFileReaderFactory::create (int *fileType*, const std::string & *fileName*, const std::string & *configFile* = "", const std::vector< std::string > & *args* = std::vector< std::string >()) [static]

Factory for CDMReader of input-files.

The function create reader tries to create a reader by filetype MIFI_FILETYPE_*. The optional arguments are defined by the different readers. Use default objects (empty string, empty vector) if arguments are not desired.

Parameters

fileNype,one of MIFI_FILETYPE_*, possibly read by detectFileType()

fileName,name of input type

configFile

options optional options for the CDMReader

Returns

pointer to CDMReader

Exceptions

CDMException if type not compiled in, or creation fails

Deprecated

use create(int fileType, const **std::string**& fileName, const XMLInput& configXML, const std::vector<std::string>& args = std::vector<std::string>())

15.15.2.5 static int MetNoFimex::CDMFileReaderFactory::detectFileType (const std::string & *fileName*) [static]

detect the filetype of a input-file

The detectFileType function uses heuristics (appendix, magic characters) to detect the filetype

Parameters

fileName input file

Returns

one of the MIFI_FILETYPE_* flags

Exceptions

if file not found

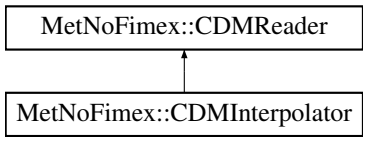
The documentation for this class was generated from the following file:

- include/fimex/CDMFileReaderFactory.h

15.16 MetNoFimex::CDMInterpolator Class Reference

`#include <CDMInterpolator.h>`

Inheritance diagram for MetNoFimex::CDMInterpolator:



Public Member Functions

- CDMInterpolator (boost::shared_ptr< CDMReader > dataReader)
- virtual ~CDMInterpolator ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos=0)

  retrieve data from the underlying dataReader and interpolate the values due to the current projection

- virtual void changeProjection (int method, const **std::string** &proj_input, const **std::vector**< double > &out_x_axis, const **std::vector**< double > &out_y_axis, const **std::string** &out_x_axis_unit, const **std::string** &out_y_axis_unit, CDMDataType out_x_axis_type, CDMDataType out_y_axis_type)
- DEPRECATED (virtual void changeProjection(int method, const **std::string** &proj_input, const **std::vector**< double > &out_x_axis, const **std::vector**< double > &out_y_axis, const **std::string** &out_x_axis_unit, const **std::string** &out_y_axis_unit))
- virtual void changeProjection (int method, const **std::string** &proj_input, const **std::string** &out_x_axis, const **std::string** &out_y_axis, const **std::string** &out_x_axis_unit, const **std::string** &out_y_axis_unit, const **std::string** &out_x_axis_type="double", const **std::string** &out_y_axis_type="double")
- virtual void changeProjection (int method, const **std::string** &netcdf_template_file)
- virtual void setLatitudeName (const **std::string** &latName)
- virtual const **std::string** & getLatitudeName () const
- virtual void setLongitudeName (const **std::string** &lonName)
- virtual const **std::string** & getLongitudeName () const
- virtual void addPreprocess (boost::shared_ptr< InterpolatorProcess2d > process)

15.16.1 Constructor & Destructor Documentation

15.16.1.1 MetNoFimex::CDMInterpolator::CDMInterpolator (boost::shared_ptr< CDMReader > *dataReader*)

15.16.1.2 virtual MetNoFimex::CDMInterpolator::~CDMInterpolator () `[virtual]`

15.16.2 Member Function Documentation

15.16.2.1 virtual void MetNoFimex::CDMInterpolator::addPreprocess (boost::shared_ptr< InterpolatorProcess2d > *process*) `[virtual]`

add a process to the internal list of preprocesses

Warning

this function is not completely thought through and might change

15.16.2.2 virtual void MetNoFimex::CDMInterpolator::changeProjection (int *method*, const std::string & *netcdf_template_file*) [virtual]

@ brief change the (main) projection of the dataReaders cdm to this new projection

Parameters

method Interpolation method

netcdf-template-file input-string for netcf template filename

15.16.2.3 virtual void MetNoFimex::CDMInterpolator::changeProjection (int *method*, const std::string & *proj_input*, const std::string & *out_x_axis*, const std::string & *out_y_axis*, const std::string & *out_x_axis_unit*, const std::string & *out_y_axis_unit*, const std::string & *out_x_axis_type* = "double", const std::string & *out_y_axis_type* = "double") [virtual]

@ brief change the (main) projection of the dataReaders cdm to this new projection

Parameters

method Interpolation method

proj_input input-string for proj4, used as output projection

out_x_axis config-string for x_axis, either '1,2,...,5' or 'auto' or 'auto,distance=3.5'

out_y_axis config-string for y_axis, either '1,2,...,5' or 'auto' or 'auto,distance=3.5'

out_x_axis_unit unit of the output x-axis

out_y_axis_unit unit of the output y-axis

out_x_axis_type type (double, float, int, short) of x-axis

out_y_axis_type type of MIFI_TYPE (double, float, int, short) of y-axis

15.16.2.4 virtual void MetNoFimex::CDMInterpolator::changeProjection (int *method*, const std::string & *proj_input*, const std::vector< double > & *out_x_axis*, const std::vector< double > & *out_y_axis*, const std::string & *out_x_axis_unit*, const std::string & *out_y_axis_unit*, CDMDataType *out_x_axis_type*, CDMDataType *out_y_axis_type*) [virtual]

@ brief change the (main) projection of the dataReaders cdm to this new projection

Parameters

method Interpolation method

proj_input input-string for proj4, used as output projection

out_x_axis values of the output x-axis

out_y_axis values of the output y-axis

out_x_axis_unit unit of the output x-axis

out_y_axis_unit unit of the output y-axis

out_x_axis_type type of CDM_TYPE (DOUBLE, FLOAT, ...) of x-axis

out_y_axis_type type of CDM_TYPE (DOUBLE, FLOAT, ...) of y-axis

15.16.2.5 MetNoFimex::CDMInterpolator::DEPRECATED (virtual void *changeProjection*int method, const std::string &proj_input, const std::vector< double > &out_x_axis, const std::vector< double > &out_y_axis, const std::string &out_x_axis_unit, const std::string &out_y_axis_unit)

@ brief change the (main) projection of the dataReaders cdm to this new projection

Parameters

method Interpolation method

proj_input input-string for proj4, used as output projection

out_x_axis values of the output x-axis

out_y_axis values of the output y-axis

out_x_axis_unit unit of the output x-axis

out_y_axis_unit unit of the output y-axis

Deprecated

use version changeProjection(int method, const std::string& proj_input, const std::vector<double>& out_x_axis, const std::vector<double>& out_y_axis, const std::string& out_x_axis_unit, const std::string& out_y_axis_unit)

15.16.2.6 virtual boost::shared_ptr<Data> MetNoFimex::CDMInterpolator::getDataSlice (const std::string & *varName*, size_t *unLimDimPos* = 0) [virtual]

retrieve data from the underlying dataReader and interpolate the values due to the current projection

Implements MetNoFimex::CDMReader.

15.16.2.7 virtual const std::string& MetNoFimex::CDMInterpolator::getLatitudeName () const [inline, virtual]

Returns

the name used for latitude in the automatic coordinate generation

15.16.2.8 virtual const std::string& MetNoFimex::CDMInterpolator::getLongitudeName () const [inline, virtual]

Returns

the name used for longitude in the automatic coordinate generation

15.16.2.9 virtual void MetNoFimex::CDMInterpolator::setLatitudeName (const std::string & *latName*) [inline, virtual]

set the name for the automatically generated latitude coordinate axis. This must be set before changeProjection is called.

Parameters

latName name for latitude

15.16.2.10 virtual void MetNoFimex::CDMInterpolator::setLongitudeName (const std::string & *lonName*) [inline, virtual]

set the name for the automatically generated longitude coordinate axis. This must be set before changeProjection is called.

Parameters

latName name for longitude

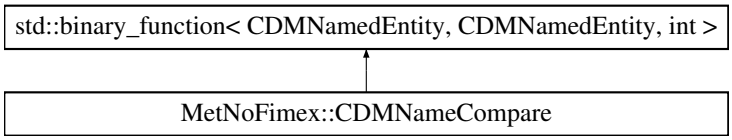
The documentation for this class was generated from the following file:

- include/fimex/CDMInterpolator.h

15.17 MetNoFimex::CDMNameCompare Struct Reference

`#include <CDMNamedEntity.h>`

Inheritance diagram for MetNoFimex::CDMNameCompare:



Public Member Functions

- int operator() (const CDMNamedEntity &e1, const CDMNamedEntity &e2)

15.17.1 Detailed Description

functor to compares names of two CDMNamedEntity using std::string::compare

15.17.2 Member Function Documentation

15.17.2.1 int MetNoFimex::CDMNameCompare::operator() (const CDMNamedEntity & *e1*, const CDMNamedEntity & *e2*) [inline]

References MetNoFimex::CDMNamedEntity::getName().

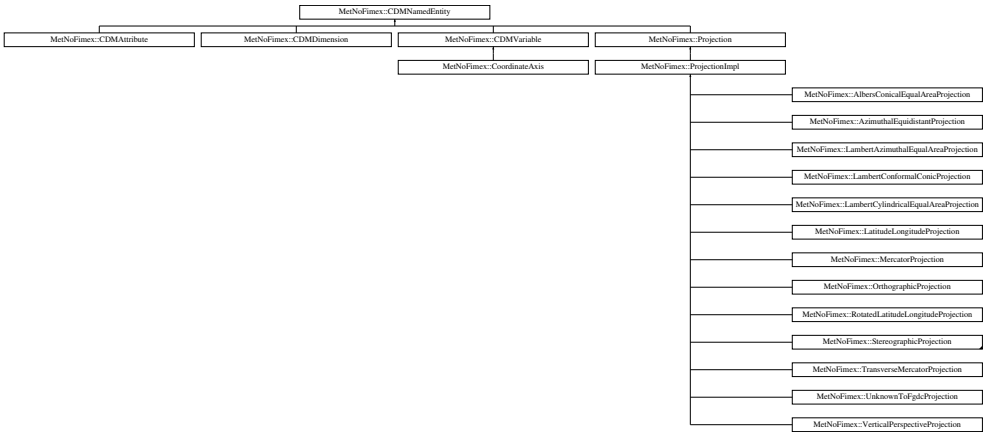
The documentation for this struct was generated from the following file:

- include/fimex/CDMNamedEntity.h

15.18 MetNoFimex::CDMNamedEntity Class Reference

`#include <CDMNamedEntity.h>`

Inheritance diagram for MetNoFimex::CDMNamedEntity:



Public Member Functions

- virtual ~CDMNamedEntity ()=0
- virtual const **std::string** & getName () const =0

15.18.1 Detailed Description

interface for all CDM Entities (variable, attribute, dimension) which support the 'getName' method

15.18.2 Constructor & Destructor Documentation

15.18.2.1 virtual MetNoFimex::CDMNamedEntity::~CDMNamedEntity () [pure virtual]

15.18.3 Member Function Documentation

15.18.3.1 virtual const std::string& MetNoFimex::CDMNamedEntity::getName () const [pure virtual]

Implemented in MetNoFimex::CDMAttribute, MetNoFimex::CDMDimension, MetNoFimex::CDMVariable, MetNoFimex::Projection, and MetNoFimex::ProjectionImpl.

Referenced by MetNoFimex::CDMNameEqual::operator()(), and MetNoFimex::CDMNameCompare::operator()().

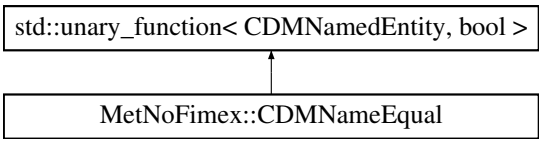
The documentation for this class was generated from the following file:

- include/fimex/CDMNamedEntity.h

15.19 MetNoFimex::CDMNameEqual Class Reference

`#include <CDMNamedEntity.h>`

Inheritance diagram for MetNoFimex::CDMNameEqual:



Public Member Functions

- CDMNameEqual (**std::string** name)
- CDMNameEqual (const CDMNamedEntity &entity)
- ~CDMNameEqual ()
- bool operator() (const CDMNamedEntity &e)

15.19.1 Detailed Description

functor to find a CDMNamedEntity equal to the set name using std::string::operator==

15.19.2 Constructor & Destructor Documentation

15.19.2.1 MetNoFimex::CDMNameEqual::CDMNameEqual (std::string *name*) [inline, explicit]

15.19.2.2 MetNoFimex::CDMNameEqual::CDMNameEqual (const CDMNamedEntity & *entity*) [inline, explicit]

15.19.2.3 MetNoFimex::CDMNameEqual::~CDMNameEqual () [inline]

15.19.3 Member Function Documentation

15.19.3.1 bool MetNoFimex::CDMNameEqual::operator() (const CDMNamedEntity & *e*) [inline]

References MetNoFimex::CDMNamedEntity::getName().

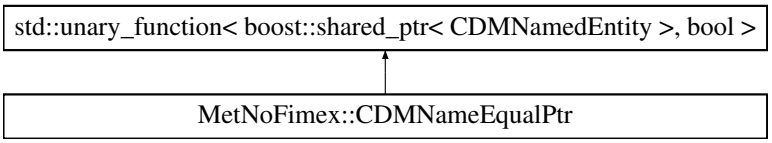
The documentation for this class was generated from the following file:

- include/fimex/CDMNamedEntity.h

15.20 MetNoFimex::CDMNameEqualPtr Class Reference

`#include <CDMNamedEntity.h>`

Inheritance diagram for MetNoFimex::CDMNameEqualPtr:



Public Member Functions

- CDMNameEqualPtr (**std::string** name)
- CDMNameEqualPtr (const boost::shared_ptr< const CDMNamedEntity > &entity)
- ~CDMNameEqualPtr ()
- bool operator() (const boost::shared_ptr< const CDMNamedEntity > &e)

15.20.1 Detailed Description

functor to find a boost::shared_ptr<CDMNamedEntity> equal to the set name using std::string::operator==

15.20.2 Constructor & Destructor Documentation

15.20.2.1 MetNoFimex::CDMNameEqualPtr::CDMNameEqualPtr (std::string *name*) [inline, explicit]

15.20.2.2 MetNoFimex::CDMNameEqualPtr::CDMNameEqualPtr (const boost::shared_ptr< const CDMNamedEntity > & *entity*) [inline, explicit]

15.20.2.3 MetNoFimex::CDMNameEqualPtr::~CDMNameEqualPtr () [inline]

15.20.3 Member Function Documentation

15.20.3.1 bool MetNoFimex::CDMNameEqualPtr::operator() (const boost::shared_ptr< const CDMNamedEntity > & *e*) [inline]

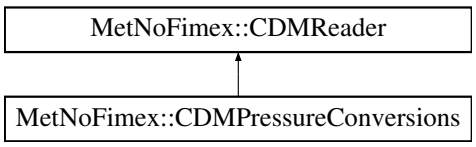
The documentation for this class was generated from the following file:

- include/fimex/CDMNamedEntity.h

15.21 MetNoFimex::CDMPressureConversions Class Reference

`#include <CDMPressureConversions.h>`

Inheritance diagram for MetNoFimex::CDMPressureConversions:



Public Member Functions

- CDMPressureConversions (boost::shared_ptr< CDMReader > dataReader, **std::vector**< **std::string** > operations)
- virtual ~CDMPressureConversions ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos=0)

  data-reading function to be called from the CDMWriter

15.21.1 Detailed Description

CDMReader to convert pressure related variables, i.e. Theta or pressure to other fields.

15.21.2 Constructor & Destructor Documentation

15.21.2.1 MetNoFimex::CDMPressureConversions::CDMPressureConversions (boost::shared_ptr< CDMReader > *dataReader*, std::vector< std::string > *operations*)

initialization with another dataReader

Parameters

dataReader source of data

operations list of operations

- theta2T translates theta (detected by standard_name) to air_temperature,
- omega2vwind will convert vertical pressure flux (omega) to vertical wind
- add4Dpressure will add variable pressure(t,k,x,y) comparable to the first 4D field found

Warning

the routine does not handle invalid values, except float/double nans

15.21.2.2 virtual MetNoFimex::CDMPressureConversions::~CDMPressureConversions () [inline, virtual]

15.21.3 Member Function Documentation

15.21.3.1 virtual boost::shared_ptr<Data> MetNoFimex::CDMPressureConversions::getDataSlice (const std::string & *varName*, size_t *unLimDimPos* = 0) [virtual]

data-reading function to be called from the CDMWriter

This methods needs to be implemented by the CDMReader. It should provide the data for each variable, either by reading from disk, converting from another CDMReader or reading from an in-memory data-section.

This function should retrieve the whole data for a dataset without unlimited dimension if the unLimDimPos == 0.

Parameters

varName name of the variable to read

unLimDimPos (optional) if the variable contains a unlimited dimension (max one allowed) an slice of this position is returned

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

Implements MetNoFimex::CDMReader.

The documentation for this class was generated from the following file:

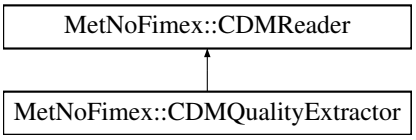
- include/fimex/CDMPressureConversions.h

15.22 MetNoFimex::CDMQualityExtractor Class Reference

Extract data with defined quality status.

`#include <CDMQualityExtractor.h>`

Inheritance diagram for MetNoFimex::CDMQualityExtractor:



Public Member Functions

- CDMQualityExtractor (boost::shared_ptr< CDMReader > dataReader, **std::string** autoConfString="", **std::string** configFile="")
- virtual ~CDMQualityExtractor ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos=0)
- const **std::map**< **std::string**, **std::string** > getStatusVariable () const
- const **std::map**< **std::string**, **std::string** > getVariableFlags () const
- const **std::map**< **std::string**, **std::vector**< double > > getVariableValues () const

15.22.1 Detailed Description

Extract data with defined quality status. The CDMQualityExtractor will select data from data-sources matching only configurable quality constraints. Data not matching these constraints will be set to undefined.

The configuration works either semi-automatic by interpreting the quality flags as given in CF-1.x at `http://cf-pcmdi.llnl.gov/documents/cf-conventions/1.4/cf-conventions.html#flags` or by using a configuration-file describing the quality-relations between the different varibles.

All variables with no quality-configuration will not be changed.

Warning

The CDMQualityExtractor will read the status-variable after applying eventual quality-flags to them. It is therefore the task of the writer of the configuration, that no circular quality-flags exist.

15.22.2 Constructor & Destructor Documentation

15.22.2.1 MetNoFimex::CDMQualityExtractor::CDMQualityExtractor (boost::shared_ptr< CDMReader > *dataReader*, std::string *autoConfString* = "", std::string *configFile* = "")

Initialize the CDMQualityExtractor

Parameters

dataReader the data-source

autoConfString the default value for CF-1.4 compatible status_flags, i.e. "all, highest, lowest, values=0,1,...,3", the values here might be overwritten by the config-file. If empty, no quality extraction on the basis of CF-1.4 will be used.

configFile filename of a cdmQualityConfig.xml file. If empty, no quality-file will be used.

Referenced by getVariableFlags().

15.22.2.2 virtual MetNoFimex::CDMQualityExtractor::~CDMQualityExtractor () [inline, virtual]

Referenced by getVariableFlags().

15.22.3 Member Function Documentation

15.22.3.1 virtual boost::shared_ptr<Data> MetNoFimex::CDMQualityExtractor::getDataSlice (const std::string & *varName*, size_t *unLimDimPos* = 0) [virtual]

Read and manipulate the data

Implements MetNoFimex::CDMReader.

Referenced by getVariableFlags().

15.22.3.2 const std::map<std::string, std::string> MetNoFimex::CDMQualityExtractor::getStatusVariable () const [inline]

Read the internals of statusVariable. This code is mainly thought for testing/debugging.

Referenced by getVariableFlags().

15.22.3.3 const std::map<std::string, std::string> MetNoFimex::CDMQualityExtractor::getVariableFlags () const [inline]

Read the internals of variableFlags, for testing/debugging.

References CDMQualityExtractor(), getDataSlice(), getStatusVariable(), getVariableValues(), and ~CDMQualityExtractor().

15.22.3.4 const std::map<std::string, std::vector<double> > MetNoFimex::CDMQualityExtractor::getVariableValues () const [inline]

Read the internals of variableValues, for testing/debugging.

Referenced by getVariableFlags().

The documentation for this class was generated from the following file:

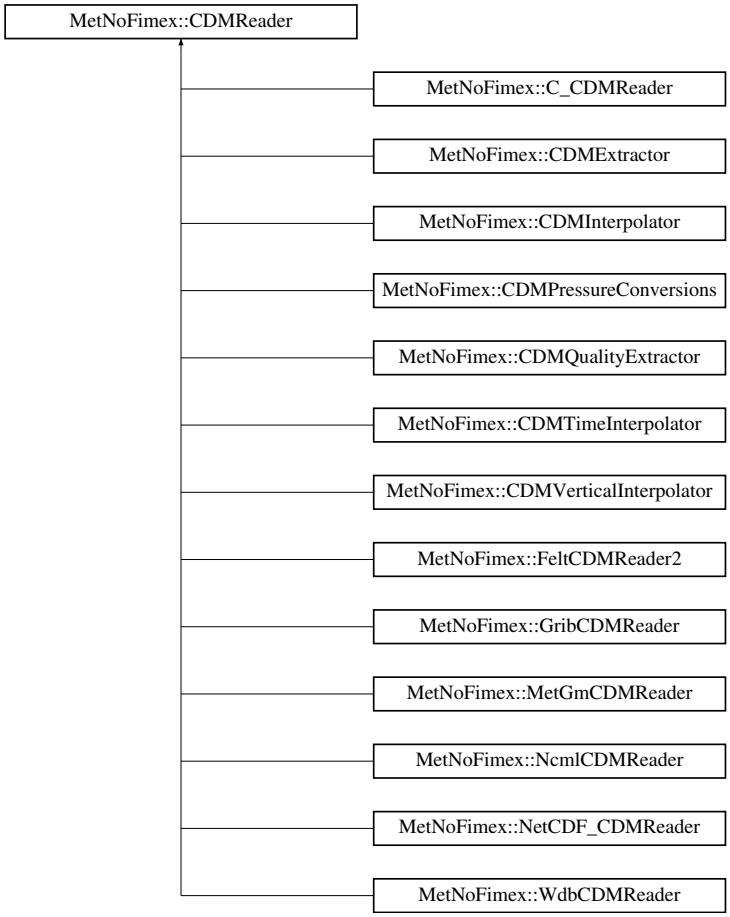
- include/fimex/CDMQualityExtractor.h

15.23 MetNoFimex::CDMReader Class Reference

Basic interface for CDM reading and manipulation classes.

`#include <CDMReader.h>`

Inheritance diagram for MetNoFimex::CDMReader:



Public Member Functions

- CDMReader ()
- virtual ~CDMReader ()
- virtual const CDM & getCDM () const
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos)=0

  data-reading function to be called from the CDMWriter

- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, const SliceBuilder &sb)

  data-reading function to be called from the CDMWriter

- virtual boost::shared_ptr< Data > getData (const **std::string** &varName)

data-reading function to be called from the CDMWriter

- virtual boost::shared_ptr< Data > getScaledDataSlice (const **std::string** &varName, size_t unLimDimPos)

  read and scale a dataslice

- virtual boost::shared_ptr< Data > getScaledDataSliceInUnit (const **std::string** &varName, const **std::string** &unit, size_t unLimDimPos)

  read and scale a dataslice to a known unit

- virtual boost::shared_ptr< Data > getScaledDataSlice (const **std::string** &varName, const SliceBuilder &sb)

  read and scale a dataslice

- virtual boost::shared_ptr< Data > getScaledDataSliceInUnit (const **std::string** &varName, const **std::string** &unit, const SliceBuilder &sb)

  read and scale a dataslice to a set unit

- virtual boost::shared_ptr< Data > getScaledData (const **std::string** &varName)

  read and scale the complete data

- virtual boost::shared_ptr< Data > getScaledDataInUnit (const **std::string** &varName, const **std::string** &unit)

  read and scale the complete data to a set unit

Protected Member Functions

- virtual boost::shared_ptr< Data > getDataSliceFromMemory (const CDMVariable &variable, size_t unLimDimPos=0)

Protected Attributes

- boost::shared_ptr< CDM > cdm_

15.23.1 Detailed Description

Basic interface for CDM reading and manipulation classes. The CDMReader is the basic interface for reading and manipulation of the cdm datastructure. The CDMWriter will work with an implementation of the CDMReader and read the included data in the cdm or the data provided through the implementation of the CDMReader#getDataSlice

See also

FeltCDMReader

15.23.2 Constructor & Destructor Documentation

15.23.2.1 MetNoFimex::CDMReader::CDMReader ()

15.23.2.2 virtual MetNoFimex::CDMReader::~CDMReader () [inline, virtual]

15.23.3 Member Function Documentation

15.23.3.1 virtual const CDM& MetNoFimex::CDMReader::getCDM () const [virtual]

Retrieve the cdm structure of this reader.

15.23.3.2 virtual boost::shared_ptr<Data> MetNoFimex::CDMReader::getData (const std::string & *varName*) [virtual]

data-reading function to be called from the CDMWriter

The getData function is a convenient function to retrieve all data from a file. It is implemented using getDataSlice. It should be used with care, since a complete variable might be bigger than available memory.

Parameters

varName name of the variable to read

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

15.23.3.3 virtual boost::shared_ptr<Data> MetNoFimex::CDMReader::getDataSlice (const std::string & *varName*, const SliceBuilder & *sb*) [virtual]

data-reading function to be called from the CDMWriter

Parameters

varName name of the variable to read

sb a SliceBuilder generated from this CDMReaders CDM

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions. This method has a default implementation depending on getDataSlice(varName, unLimDimPos), but should be implemented for performance reasons.

Reimplemented in MetNoFimex::MetGmCDMReader, MetNoFimex::NetCDF_CDMReader, and MetNoFimex::WdbCDMReader.

15.23.3.4 virtual boost::shared_ptr<Data> MetNoFimex::CDMReader::getDataSlice (const std::string & *varName*, size_t *unLimDimPos*) [pure virtual]

data-reading function to be called from the CDMWriter

This methods needs to be implemented by the CDMReader. It should provide the data for each variable, either by reading from disk, converting from another CDMReader or reading from an in-memory data-section.

This function should retrieve the whole data for a dataset without unlimited dimension if the unLimDimPos == 0.

Parameters

varName name of the variable to read

unLimDimPos (optional) if the variable contains a unlimited dimension (max one allowed) an slice of this position is returned

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

Implemented in MetNoFimex::C_CDMReader, MetNoFimex::CDMExtractor, MetNoFimex::CDMInterpolator, MetNoFimex::CDMPressureConversions, MetNoFimex::CDMQualityExtractor, MetNoFimex::CDMTimeInterpolator, MetNoFimex::CDMVerticalInterpolator, MetNoFimex::FeltCDMReader2, MetNoFimex::GribCDMReader, MetNoFimex::MetGmCDMReader, MetNoFimex::NcmlCDMReader, MetNoFimex::NetCDF_CDMReader, and MetNoFimex::WdbCDMReader.

15.23.3.5 virtual boost::shared_ptr<Data> MetNoFimex::CDMReader::getDataSliceFromMemory (const CDMVariable & *variable*, size_t *unLimDimPos* = 0) [protected, virtual]

Read the data from the variable.hasData() and select the correct unLimDimPos. This function should be used internally from getDataSlice.

Parameters

variable the variable to read data from

unLimDimPos (optional) the unlimited position

15.23.3.6 virtual boost::shared_ptr<Data> MetNoFimex::CDMReader::getScaledData (const std::string & *varName*) [virtual]

read and scale the complete data

This functions uses getData internally. It tries to read "scale_factor" "add_offset" and "_FillValue" and apply the scaling to the read data. Output-datatype will be double, output _FillValue will be MIFI_UNDEFINED_D.

Parameters

varName name of the variable to read

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

15.23.3.7 virtual boost::shared_ptr<Data> MetNoFimex::CDMReader::getScaledDataInUnit (const std::string & *varName*, const std::string & *unit*) [virtual]

read and scale the complete data to a set unit

This functions uses getData internally. It tries to read "scale_factor" "add_offset" and "_FillValue" and apply the scaling to the read data. Output-datatype will be double, output _FillValue will be MIFI_-UNDEFINED_D. The data will be converted to match unit.

Parameters

varName name of the variable to read

unit the unit-string to convert the data to

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

15.23.3.8 virtual boost::shared_ptr<Data> MetNoFimex::CDMReader::getScaledDataSlice (const std::string & *varName*, const SliceBuilder & *sb*) [virtual]

read and scale a dataslice

Parameters

varName name of the variable to read

sb SliceBuilder to restrict the data

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

See also

getScaledDataSlice(varName, unLimDimPos)

15.23.3.9 virtual boost::shared_ptr<Data> MetNoFimex::CDMReader::getScaledDataSlice (const std::string & *varName*, size_t *unLimDimPos*) [virtual]

read and scale a dataslice

This functions uses getDataSlice internally. It tries to read "scale_factor" "add_offset" and "_FillValue" and apply the scaling to the read data. Output-datatype will be double, output _FillValue will be MIFI_-UNDEFINED_D

Parameters

varName name of the variable to read

unLimDimPos (optional) if the variable contains a unlimited dimension (max one allowed) an slice of this position is returned

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

15.23.3.10 virtual boost::shared_ptr<Data> MetNoFimex::CDMReader::getScaledDataSliceInUnit (const std::string & *varName*, const std::string & *unit*, const SliceBuilder & *sb*) [virtual]

read and scale a dataslice to a set unit

Parameters

varName name of the variable to read

unit unit string to scale to

sb SliceBuilder to restrict the data

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

See also

getScaledDataSlice(varName, unLimDimPos)

15.23.3.11 virtual boost::shared_ptr<Data> MetNoFimex::CDMReader::getScaledDataSliceInUnit (const std::string & *varName*, const std::string & *unit*, size_t *unLimDimPos*) [virtual]

read and scale a dataslice to a known unit

This functions uses getDataSlice internally. It tries to read "scale_factor" "add_offset" and "_FillValue" and apply the scaling to the read data. Output-datatype will be double, output _FillValue will be MIFI_UNDEFINED_D. The data will be converted to match unit.

Parameters

varName name of the variable to read

unit unit-string

unLimDimPos (optional) if the variable contains a unlimited dimension (max one allowed) an slice of this position is returned

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

15.23.4 Member Data Documentation

15.23.4.1 boost::shared_ptr<CDM> MetNoFimex::CDMReader::cdm_ [protected]

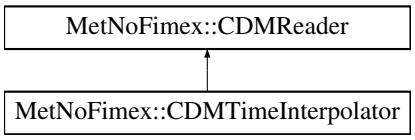
The documentation for this class was generated from the following file:

- include/fimex/CDMReader.h

15.24 MetNoFimex::CDMTimeInterpolator Class Reference

`#include <CDMTimeInterpolator.h>`

Inheritance diagram for MetNoFimex::CDMTimeInterpolator:



Public Member Functions

- CDMTimeInterpolator (boost::shared_ptr< CDMReader > dataReader)
- virtual ~CDMTimeInterpolator ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos=0)

  retrieve data from the underlying dataReader and interpolate the values due to the current projection

- virtual void changeTimeAxis (**std::string** timeSpec)

15.24.1 Constructor & Destructor Documentation

15.24.1.1 MetNoFimex::CDMTimeInterpolator::CDMTimeInterpolator (boost::shared_ptr< CDMReader > *dataReader*)

15.24.1.2 virtual MetNoFimex::CDMTimeInterpolator::~CDMTimeInterpolator () `[virtual]`

15.24.2 Member Function Documentation

15.24.2.1 virtual void MetNoFimex::CDMTimeInterpolator::changeTimeAxis (std::string *timeSpec*) `[virtual]`

change the time-axis from from the one given to a new specification

Parameters

timeSpec string of time-specification

Exceptions

CDMException on unparsable timeSpec

See also

secTimeSpec

15.24.2.2 virtual boost::shared_ptr<Data> MetNoFimex::CDMTimeInterpolator::getDataSlice (const std::string & *varName*, size_t *unLimDimPos* = 0) [virtual]

retrieve data from the underlying dataReader and interpolate the values due to the current projection

Parameters

varName name of variable

size_t unLimDimPos position of the unlimited dimension, most commonly time-position of the output as set in changeTimeAxis

Implements MetNoFimex::CDMReader.

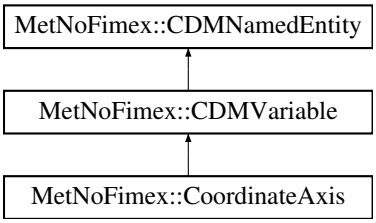
The documentation for this class was generated from the following file:

- include/fimex/CDMTimeInterpolator.h

15.25 MetNoFimex::CDMVariable Class Reference

`#include "fimex/CDMVariable.h"`

Inheritance diagram for MetNoFimex::CDMVariable:



Public Member Functions

- CDMVariable (**std::string** name, CDMDataType datatype, **std::vector**< **std::string** > shape)
- virtual ~CDMVariable ()
- const **std::string** & getName () const
- void setName (**std::string** newName)
- CDMDataType getDataType () const
- void setDataType (CDMDataType type)
- const **std::vector**< **std::string** > & getShape () const
- void setShape (**std::vector**< **std::string** > newShape)
- void setAsSpatialVector (const **std::string** &counterpart, const **std::string** &direction)
- bool isSpatialVector () const
- const **std::string** & getSpatialVectorCounterpart () const

  get the spatial counterpart of this vector

- const **std::string** & getSpatialVectorDirection () const

  get the possible directions of this spatial vector (comma-separated string)

- bool checkDimension (const **std::string** &dimension) const
- void toXMLStream (**std::ostream** &out) const

  print a xml representation to the stream without attributes

- void toXMLStream (**std::ostream** &out, const **std::vector**< CDMAttribute > &attrs) const

  print a xml representation to the stream with attributes

- void setData (boost::shared_ptr< Data > data)

  add data to the variable

- const boost::shared_ptr< Data > getData () const

  retrieve volatile data from this variable

- int hasData () const

  check if real data has been set with setData() (null-pointer reference returns false)

15.25.1 Constructor & Destructor Documentation

15.25.1.1 MetNoFimex::CDMVariable::CDMVariable (std::string *name*, CDMDataType *datatype*, std::vector< std::string > *shape*) [explicit]

15.25.1.2 virtual MetNoFimex::CDMVariable::~CDMVariable () [virtual]

15.25.2 Member Function Documentation

15.25.2.1 bool MetNoFimex::CDMVariable::checkDimension (const std::string & *dimension*) const

check the dimension of a variable

Parameters

dimension the dimension to check for

15.25.2.2 const boost::shared_ptr<Data> MetNoFimex::CDMVariable::getData () const [inline]

retrieve volatile data from this variable

Retrieve data, but only if it has been set previously by setData() this method will not try to read data from the disk. Use CDMReader::getData(const std::string& varName) to get the data from memory or from disk.

15.25.2.3 CDMDataType MetNoFimex::CDMVariable::getDataType () const [inline]

15.25.2.4 const std::string& MetNoFimex::CDMVariable::getName () const [inline, virtual]

Implements MetNoFimex::CDMNamedEntity.

Referenced by MetNoFimex::CoordinateAxis::operator<().

15.25.2.5 const std::vector<std::string>& MetNoFimex::CDMVariable::getShape () const [inline]

15.25.2.6 const std::string& MetNoFimex::CDMVariable::getSpatialVectorCounterpart () const [inline]

get the spatial counterpart of this vector

15.25.2.7 const std::string& MetNoFimex::CDMVariable::getSpatialVectorDirection () const [inline]

get the possible directions of this spatial vector (comma-separated string)

15.25.2.8 int MetNoFimex::CDMVariable::hasData () const [inline]

check if real data has been set with setData() (null-pointer reference returns false)

15.25.2.9 bool MetNoFimex::CDMVariable::isSpatialVector () const [inline]

test if this variable has been declared to be a spatial vector

15.25.2.10 void MetNoFimex::CDMVariable::setAsSpatialVector (const std::string & *counterpart*, const std::string & *direction*)

Declare this variable to be part of a spatial vector, e.g. (x-wind, y-wind)

Parameters

counterpart name of the other variable being part of this vector

direction comma-separated list of possible directions for this vector, e.g. "x,longitude"

15.25.2.11 void MetNoFimex::CDMVariable::setData (boost::shared_ptr< Data > *data*) [inline]

add data to the variable

15.25.2.12 void MetNoFimex::CDMVariable::setDataType (CDMDataType *type*) [inline]

15.25.2.13 void MetNoFimex::CDMVariable::setName (std::string *newName*) [inline]

15.25.2.14 void MetNoFimex::CDMVariable::setShape (std::vector< std::string > *newShape*) [inline]

15.25.2.15 void MetNoFimex::CDMVariable::toXMLStream (std::ostream & *out*, const std::vector< CDMAttribute > & *attrs*) const

print a xml representation to the stream with attributes

15.25.2.16 void MetNoFimex::CDMVariable::toXMLStream (std::ostream & *out*) const

print a xml representation to the stream without attributes

The documentation for this class was generated from the following file:

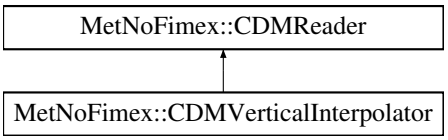
- include/fimex/CDMVariable.h

15.26 MetNoFimex::CDMVerticalInterpolator Class Reference

Interpolation of vertical layers.

`#include <CDMVerticalInterpolator.h>`

Inheritance diagram for MetNoFimex::CDMVerticalInterpolator:



Public Member Functions

- CDMVerticalInterpolator (boost::shared_ptr< CDMReader > dataReader, **std::string** verticalType, **std::string** verticalInterpolationMethod, const **std::vector**< double > level1, const **std::vector**< double > level2)
- virtual ~CDMVerticalInterpolator ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos=0)

Static Public Member Functions

- static void getSimpleAxes (const boost::shared_ptr< const CoordinateSystem > &cs, const CDM &cdm, CoordinateSystem::ConstAxisPtr &xAxis, CoordinateSystem::ConstAxisPtr &yAxis, CoordinateSystem::ConstAxisPtr &zAxis, CoordinateSystem::ConstAxisPtr &tAxis, size_t &nx, size_t &ny, size_t &nz, size_t &nt, bool &tIsUnlimited)

15.26.1 Detailed Description

Interpolation of vertical layers. The CDMVerticalInterpolator can be used to interpolate vertical levels.

Warning

CDMVerticalInterpolator requires a valid coordinate-system. In addition, the vertical axis may not be the unlimited dimension. Furthermore, the vertical layer may not depend or more dimensions than x,y and time. The order of dimensions must be time, k, y, x.
The routine does not handle invalid values, except float/double nans

15.26.2 Constructor & Destructor Documentation

15.26.2.1 MetNoFimex::CDMVerticalInterpolator::CDMVerticalInterpolator (boost::shared_ptr< CDMReader > *dataReader*, std::string *verticalType*, std::string *verticalInterpolationMethod*, const std::vector< double > *level1*, const std::vector< double > *level2*)

Initialize a vertical interpolator.

Parameters

dataReader the data-source

verticalType must be 'pressure'

interpolationMethod one of 'linear', 'log', 'loglog'

level1 the new vertical levels, for 'pressure', that is pressure in hPa

level2 only required for hybrid levels, not yet supported

15.26.2.2 virtual MetNoFimex::CDMVerticalInterpolator::~CDMVerticalInterpolator () [virtual]

15.26.3 Member Function Documentation

15.26.3.1 virtual boost::shared_ptr<Data> MetNoFimex::CDMVerticalInterpolator::getDataSlice (const std::string & *varName*, size_t *unLimDimPos* = 0) [virtual]

retrieve data from the underlying dataReader and interpolate the values to the new vertical levels

Implements MetNoFimex::CDMReader.

15.26.3.2 static void MetNoFimex::CDMVerticalInterpolator::getSimpleAxes (const boost::shared_ptr< const CoordinateSystem > & *cs*, const CDM & *cdm*, CoordinateSystem::ConstAxisPtr & *xAxis*, CoordinateSystem::ConstAxisPtr & *yAxis*, CoordinateSystem::ConstAxisPtr & *zAxis*, CoordinateSystem::ConstAxisPtr & *tAxis*, size_t & *nx*, size_t & *ny*, size_t & *nz*, size_t & *nt*, bool & *tIsUnlimited*) [static]

Get the axes of a simple (1-dim x,y,z,t) coordinate-system. The t-axis might be omitted or unlimited, the order of the axes must be as written above (x,y order might be reversed).

Parameters

cs the coordinate system

cdm the corresponding data model

tIsUnlmited indicate if t-axis is unlimited axes

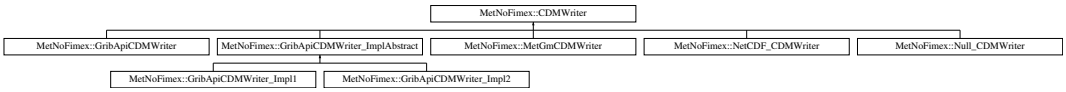
The documentation for this class was generated from the following file:

- include/fimex/CDMVerticalInterpolator.h

15.27 MetNoFimex::CDMWriter Class Reference

`#include <CDMWriter.h>`

Inheritance diagram for MetNoFimex::CDMWriter:



Public Member Functions

- CDMWriter (boost::shared_ptr< CDMReader > cdmReader, const **std::string** &outputFile)
- virtual ~CDMWriter ()

Protected Attributes

- boost::shared_ptr< CDMReader > cdmReader
- const **std::string** outputFile

15.27.1 Constructor & Destructor Documentation

15.27.1.1 MetNoFimex::CDMWriter::CDMWriter (boost::shared_ptr< CDMReader > *cdmReader*, const std::string & *outputFile*) [inline]

15.27.1.2 virtual MetNoFimex::CDMWriter::~CDMWriter () [inline, virtual]

15.27.2 Member Data Documentation

15.27.2.1 boost::shared_ptr<CDMReader> MetNoFimex::CDMWriter::cdmReader [protected]

15.27.2.2 const std::string MetNoFimex::CDMWriter::outputFile [protected]

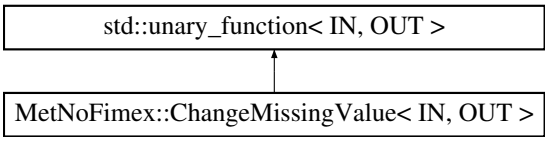
The documentation for this class was generated from the following file:

- include/fimex/CDMWriter.h

15.28 MetNoFimex::ChangeMissingValue< IN, OUT > Class Template Reference

`#include <Utils.h>`

Inheritance diagram for MetNoFimex::ChangeMissingValue< IN, OUT >:



Public Member Functions

- ChangeMissingValue (double oldFill, double newFill)
- OUT operator() (const IN &in) const

15.28.1 Detailed Description

template<typename IN, typename OUT> class MetNoFimex::ChangeMissingValue< IN, OUT >

Change the missing value

15.28.2 Constructor & Destructor Documentation

15.28.2.1 template<typename IN , typename OUT > MetNoFimex::ChangeMissingValue< IN, OUT >::ChangeMissingValue (double *oldFill*, double *newFill*) [inline]

15.28.3 Member Function Documentation

15.28.3.1 template<typename IN , typename OUT > OUT MetNoFimex::ChangeMissingValue< IN, OUT >::operator() (const IN & *in*) const [inline]

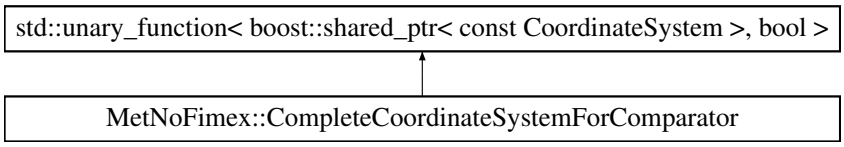
The documentation for this class was generated from the following file:

- include/fimex/Utils.h

15.29 MetNoFimex::CompleteCoordinateSystemForComparator Struct Reference

`#include <CoordinateSystem.h>`

Inheritance diagram for MetNoFimex::CompleteCoordinateSystemForComparator:



Public Member Functions

- CompleteCoordinateSystemForComparator (const **std::string** &varName)
- virtual ~CompleteCoordinateSystemForComparator ()
- bool operator() (const boost::shared_ptr< const CoordinateSystem > &cs)

15.29.1 Detailed Description

Functor to check if a coordinate system completely describes a variable, i.e. all axes match fully.

Examples:

coordinateSystem.cpp.

15.29.2 Constructor & Destructor Documentation

15.29.2.1 MetNoFimex::CompleteCoordinateSystemForComparator::CompleteCoordinateSystemForComparator (const std::string & *varName*) [inline]

15.29.2.2 virtual MetNoFimex::CompleteCoordinateSystemForComparator::~CompleteCoordinateSystemForComparator () [inline, virtual]

15.29.3 Member Function Documentation

15.29.3.1 bool MetNoFimex::CompleteCoordinateSystemForComparator::operator() (const boost::shared_ptr< const CoordinateSystem > & *cs*) [inline]

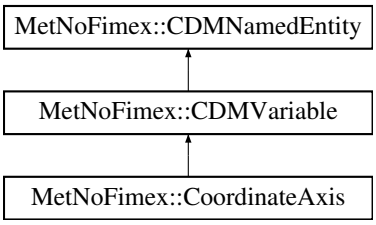
The documentation for this struct was generated from the following file:

- include/fimex/coordSys/CoordinateSystem.h

15.30 MetNoFimex::CoordinateAxis Class Reference

`#include "fimex/coordSys/CoordinateAxis.h"`

Inheritance diagram for MetNoFimex::CoordinateAxis:



Public Types

- enum AxisType {
  Undefined = 0, GeoX, GeoY, GeoZ,
  Time, Lon, Lat, Pressure,
  Height, ReferenceTime }

Public Member Functions

- CoordinateAxis (const CDMVariable &var)
- virtual ~CoordinateAxis ()
- bool operator< (const CoordinateAxis &ca)
- AxisType getAxisType () const
- **std::string** getAxisTypeStr () const
- void setAxisType (AxisType t)
- bool isAxisType (AxisType t) const
- bool isExplicit () const
- void setExplicit (bool isExplicit)

Static Public Member Functions

- static **std::string** type2string (AxisType type)

15.30.1 Member Enumeration Documentation

15.30.1.1 enum MetNoFimex::CoordinateAxis::AxisType

Enumerator:

Undefined

GeoX

GeoY

GeoZ

Time
Lon
Lat
Pressure
Height
ReferenceTime

15.30.2 Constructor & Destructor Documentation

15.30.2.1 MetNoFimex::CoordinateAxis::CoordinateAxis (const CDMVariable & *var*) [inline, explicit]

15.30.2.2 virtual MetNoFimex::CoordinateAxis::~CoordinateAxis () [inline, virtual]

15.30.3 Member Function Documentation

15.30.3.1 AxisType MetNoFimex::CoordinateAxis::getAxisType () const [inline]

15.30.3.2 std::string MetNoFimex::CoordinateAxis::getAxisTypeStr () const [inline]

References type2string().

15.30.3.3 bool MetNoFimex::CoordinateAxis::isAxisType (AxisType *t*) const [inline]

15.30.3.4 bool MetNoFimex::CoordinateAxis::isExplicit () const [inline]

Check if this axis is a explicitly netcdf-dimension, too.

Returns

false if this is a implicit 'coordinates' variable, rather than a netcdf-dimension

15.30.3.5 bool MetNoFimex::CoordinateAxis::operator< (const CoordinateAxis & *ca*) [inline]

References MetNoFimex::CDMVariable::getName().

15.30.3.6 void MetNoFimex::CoordinateAxis::setAxisType (AxisType *t*) [inline]

15.30.3.7 void MetNoFimex::CoordinateAxis::setExplicit (bool *isExplicit*) [inline]

15.30.3.8 static std::string MetNoFimex::CoordinateAxis::type2string (AxisType *type*) [inline, static]

References GeoX, GeoY, GeoZ, Height, Lat, Lon, Pressure, ReferenceTime, and Time.

Referenced by getAxisTypeStr().

The documentation for this class was generated from the following file:

- include/fimex/coordSys/CoordinateAxis.h

15.31 MetNoFimex::CoordinateSystem Class Reference

`#include "fimex/coordSys/CoordinateSystem.h"`

Public Types

- typedef boost::shared_ptr< const CoordinateAxis > ConstAxisPtr
- typedef boost::shared_ptr< CoordinateAxis > AxisPtr
- typedef **std::vector**< ConstAxisPtr > ConstAxisList

Public Member Functions

- CoordinateSystem ()
- CoordinateSystem (const **std::string** &conventionName)
- virtual ~CoordinateSystem ()
- virtual **std::string** id () const
- virtual **std::string** getConventionName () const
- virtual void setConventionName (const **std::string** &conventionName)
- virtual bool isComplete (const **std::string** &varName) const
- virtual void setComplete (const **std::string** &varName, bool set=true)
- virtual bool isCSFor (const **std::string** &varName) const
- virtual void setCSFor (const **std::string** &varName, bool set=true)
- virtual bool isSimpleSpatialGridded () const
- virtual void setSimpleSpatialGridded (bool set=true)
- virtual bool hasProjection () const
- virtual boost::shared_ptr< const Projection > getProjection () const
- virtual void setProjection (boost::shared_ptr< const Projection > proj)
- virtual bool hasAxisType (CoordinateAxis::AxisType type) const
- virtual ConstAxisPtr findAxisOfType (CoordinateAxis::AxisType type) const
- virtual ConstAxisPtr findAxisOfType (const **std::vector**< CoordinateAxis::AxisType > &types) const
- virtual ConstAxisPtr getGeoXAxis () const
- virtual ConstAxisPtr getGeoYAxis () const
- virtual ConstAxisPtr getGeoZAxis () const
- virtual ConstAxisPtr getTimeAxis () const
- virtual **ConstAxisList** getAxes () const
- virtual void setAxis (ConstAxisPtr axis)

15.31.1 Detailed Description

CoordinateSystems are usually created using the listCoordinateSystems(const CDM& cdm) function, see example there.

To investigate the coordinate systems of a file, use {*fimex* --printCS}.

15.31.2 Member Typedef Documentation

15.31.2.1 typedef boost::shared_ptr<CoordinateAxis> MetNoFimex::CoordinateSystem::AxisPtr

a garbage collected pointer to a CoordinateAxis

15.31.2.2 typedef std::vector<ConstAxisPtr> MetNoFimex::CoordinateSystem::ConstAxisList

a list to constant axis pointer

15.31.2.3 typedef boost::shared_ptr<const CoordinateAxis> MetNoFimex::CoordinateSystem::ConstAxisPtr

a garbage collected pointer to a constant coordinateAxis

15.31.3 Constructor & Destructor Documentation

15.31.3.1 MetNoFimex::CoordinateSystem::CoordinateSystem ()

CoordinateSystems are usually created within the listCoordinateSystems(const CDM& cdm) funcion.

15.31.3.2 MetNoFimex::CoordinateSystem::CoordinateSystem (const std::string & *conventionName*) `[explicit]`

15.31.3.3 virtual MetNoFimex::CoordinateSystem::~CoordinateSystem () `[inline, virtual]`

15.31.4 Member Function Documentation

15.31.4.1 virtual ConstAxisPtr MetNoFimex::CoordinateSystem::findAxisOfType (const std::vector< CoordinateAxis::AxisType > & *types*) const `[virtual]`

find the first axis with one of the types

Parameters

types list of types

Returns

an axis or null

15.31.4.2 virtual ConstAxisPtr MetNoFimex::CoordinateSystem::findAxisOfType (CoordinateAxis::AxisType *type*) const `[virtual]`

find the first axis with exactly the types

Parameters

type

Returns

an axis or null

15.31.4.3 virtual ConstAxisList MetNoFimex::CoordinateSystem::getAxes () const [virtual]

get all axes

15.31.4.4 virtual std::string MetNoFimex::CoordinateSystem::getConventionName () const [virtual]

Returns

the name of convention used to build the CS

15.31.4.5 virtual ConstAxisPtr MetNoFimex::CoordinateSystem::getGeoXAxis () const [virtual]

get the geographical x/lon-axis, that is one of GeoX, Longitude (in that order if several exist)

Returns

an axis or null

15.31.4.6 virtual ConstAxisPtr MetNoFimex::CoordinateSystem::getGeoYAxis () const [virtual]

Set/overwrite the geographic y axis, that is one of GeoY, Latitude (in that order if several exist)

Returns

an axis or null

15.31.4.7 virtual ConstAxisPtr MetNoFimex::CoordinateSystem::getGeoZAxis () const [virtual]

get the geographical z-axis, that is one of GeoZ, Height, Pressure (in that order if several exist)

Returns

an axis, or null

15.31.4.8 virtual boost::shared_ptr<const Projection> MetNoFimex::CoordinateSystem::getProjection () const [virtual]

Get the projection of the coordinate-system (projection of GeoX, GeoY and optionally GeoZ) This includes also coordinate-systems in latitude-longitude 'projection'.

Returns

projection, or null ptr

15.31.4.9 virtual ConstAxisPtr MetNoFimex::CoordinateSystem::getTimeAxis () const [virtual]

get the time-axis, or NULL/0

15.31.4.10 virtual bool MetNoFimex::CoordinateSystem::hasAxisType (CoordinateAxis::AxisType *type*) const [virtual]

Check if the CoordinateSystem contains exactly the axis type

Parameters

type axis type to check against

15.31.4.11 virtual bool MetNoFimex::CoordinateSystem::hasProjection () const [virtual]

Check if the coordinate-system has a projection (of GeoX, GeoY and optionally GeoZ) This includes also coordinate-systems in latitude-longitude 'projection'.

15.31.4.12 virtual std::string MetNoFimex::CoordinateSystem::id () const [virtual]

unique identifier for a coordinate system

15.31.4.13 virtual bool MetNoFimex::CoordinateSystem::isComplete (const std::string & *varName*) const [virtual]

All axes of this system are used by the variable varName

Parameters

varName variable name

15.31.4.14 virtual bool MetNoFimex::CoordinateSystem::isCSFor (const std::string & *varName*) const [virtual]

All dimensions of the variable are described by CoordinateSystem

Parameters

varName variable name

15.31.4.15 virtual bool MetNoFimex::CoordinateSystem::isSimpleSpatialGridded () const [virtual]

Check if coordinate system has direct spatial axes, i.e. 1-dim x,y axes or 1-dim lon,lat axes

15.31.4.16 virtual void MetNoFimex::CoordinateSystem::setAxis (ConstAxisPtr *axis*) [virtual]

Set any axis.

Exceptions

CDMException if an axis with the same axistype (except undefined) exists

15.31.4.17 virtual void MetNoFimex::CoordinateSystem::setComplete (const std::string & *varName*, bool *set* = true) [virtual]

set or unset if the coordinate system is complete for the variable

15.31.4.18 virtual void MetNoFimex::CoordinateSystem::setConventionName (const std::string & *conventionName*) [virtual]

set the convention name

Parameters

conventionName

15.31.4.19 virtual void MetNoFimex::CoordinateSystem::setCSFor (const std::string & *varName*, bool *set* = true) [virtual]

Set or unset if all dimensions are described by the CoordinateSystem

15.31.4.20 virtual void MetNoFimex::CoordinateSystem::setProjection (boost::shared_ptr< const Projection > *proj*) [virtual]

Set the projection of the coordinate-system (projection of GeoX, GeoY and optionally GeoZ)

15.31.4.21 virtual void MetNoFimex::CoordinateSystem::setSimpleSpatialGridded (bool *set* = true) [virtual]

Set or unset if this coordinate system has spatial axes, i.e. 1-dim x,y or lon/lat axes

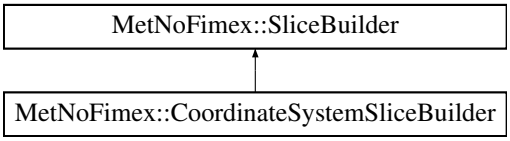
The documentation for this class was generated from the following file:

- include/fimex/coordSys/CoordinateSystem.h

15.32 MetNoFimex::CoordinateSystemSliceBuilder Class Reference

`#include <CoordinateSystemSliceBuilder.h>`

Inheritance diagram for MetNoFimex::CoordinateSystemSliceBuilder:



Public Member Functions

- CoordinateSystemSliceBuilder (const CDM &cdm, boost::shared_ptr< const CoordinateSystem > cs)
- virtual ~CoordinateSystemSliceBuilder ()
- void setReferenceTimePos (size_t refTimePos)
- void setTimeStartAndSize (size_t start, size_t size)
- SliceBuilder getTimeVariableSliceBuilder ()

15.32.1 Detailed Description

SliceBuilder with knowledge about the coordinate-system. It is therefore possible to set some variable on the basis that they are i.e. time-axes.

Examples:

coordinateSystem.cpp.

15.32.2 Constructor & Destructor Documentation

15.32.2.1 MetNoFimex::CoordinateSystemSliceBuilder::CoordinateSystemSliceBuilder (const CDM & *cdm*, boost::shared_ptr< const CoordinateSystem > *cs*)

15.32.2.2 virtual MetNoFimex::CoordinateSystemSliceBuilder::~CoordinateSystemSliceBuilder () `[inline, virtual]`

15.32.3 Member Function Documentation

15.32.3.1 SliceBuilder MetNoFimex::CoordinateSystemSliceBuilder::getTimeVariableSliceBuilder ()

Get a slice-builder to fetch data for the time-variable with the same reference-time as set for the current slice. It should be used as:

```
reader->getDataSlice(cs->getTimeAxis()->getName(), cssb->getTimeVariableSliceBui
    lder)
```

Examples:

coordinateSystem.cpp.

15.32.3.2 void MetNoFimex::CoordinateSystemSliceBuilder::setReferenceTimePos (size_t *refTimePos*)

Set a single reference-time. The CoordinateSystemSliceBuilder will only fetch a single reference-time, by default the first one.

Examples:

coordinateSystem.cpp.

15.32.3.3 void MetNoFimex::CoordinateSystemSliceBuilder::setTimeStartAndSize (size_t *start*, size_t *size*)

Set the start and the size of the time-dimension. This might even be a 2-dimensional time-dimension, i.e. (refTime,offset)

Examples:

coordinateSystem.cpp.

The documentation for this class was generated from the following file:

- include/fimex/CoordinateSystemSliceBuilder.h

15.33 MetNoFimex::Data Class Reference

`#include "fimex/Data.h"`

Public Member Functions

- virtual ~Data ()=0
- virtual size_t size () const =0

  size of the data

- virtual int bytes_for_one () const =0

  sizeof the data-impl datatype

- virtual void * getDataPtr ()=0
- virtual void toStream (**std::ostream** &, **std::string** separator="") const =0

  printing of the current data to ostream, with optional separator

- virtual boost::shared_array< const char > asConstChar () const =0

  retrieve data as char

- virtual boost::shared_array< char > asChar ()=0

  retrieve data as char

- virtual boost::shared_array< const short > asConstShort () const =0

  retrieve data as short

- virtual boost::shared_array< short > asShort ()=0

  retrieve data as short

- virtual boost::shared_array< const int > asConstInt () const =0

  retrieve data as int

- virtual boost::shared_array< int > asInt ()=0

  retrieve data as int

- virtual boost::shared_array< const long long > asConstInt64 () const =0

  retrieve data as int64

- virtual boost::shared_array< long long > asInt64 ()=0

  retrieve data as int64

- virtual boost::shared_array< const unsigned char > asConstUChar () const =0

  retrieve data as uchar

- virtual boost::shared_array< unsigned char > asUChar ()=0

  retrieve data as uchar

- virtual boost::shared_array< const unsigned short > asConstUShort () const =0

  retrieve data as short

- virtual boost::shared_array< unsigned short > asUShort ()=0

  retrieve data as short

- virtual boost::shared_array< const unsigned int > asConstUInt () const =0

  retrieve data as uint

- virtual boost::shared_array< unsigned int > asUInt ()=0

  retrieve data as uint

- virtual boost::shared_array< const unsigned long long > asConstUInt64 () const =0

  retrieve data as uint64

- virtual boost::shared_array< unsigned long long > asUInt64 ()=0

  retrieve data as uint64

- virtual boost::shared_array< const float > asConstFloat () const =0

  retrieve data as float

- virtual boost::shared_array< float > asFloat ()=0

  retrieve data as float (eventually copy)

- virtual boost::shared_array< const double > asConstDouble () const =0

  retrieve data as double

- virtual boost::shared_array< double > asDouble ()=0

  retrieve data as double

- virtual **std::string** asString (**std::string** separator="") const =0

  retrieve the whole array as a string (with possible separator)

- virtual void setValue (long pos, double val)=0

  set a value at the desired position

- virtual void setValues (size_t startPos, const Data &data, size_t first=0, size_t end=-1)=0
- virtual void setAllValues (double val)=0
- virtual boost::shared_ptr< Data > clone () const =0

  duplicate the data

- virtual boost::shared_ptr< Data > slice (**std::vector**< size_t > orgDimSize, **std::vector**< size_t > startDims, **std::vector**< size_t > outputDimSize)=0

  get a multi-dimensional slice of the data

- virtual boost::shared_ptr< Data > convertDataType (double oldFill, double oldScale, double oldOffset, CDMDataType newType, double newFill, double newScale, double newOffset)=0

  convert the datatype from one type,fill,scale,offset to another

- virtual CDMDataType getDataType () const =0

15.33.1 Detailed Description

General class for storing different basic array pointers plus length

15.33.2 Constructor & Destructor Documentation

15.33.2.1 virtual MetNoFimex::Data::~Data () [pure virtual]

15.33.3 Member Function Documentation

15.33.3.1 virtual boost::shared_array<char> MetNoFimex::Data::asChar () [pure virtual]

retrieve data as char

15.33.3.2 virtual boost::shared_array<const char> MetNoFimex::Data::asConstChar () const [pure virtual]

retrieve data as char

15.33.3.3 virtual boost::shared_array<const double> MetNoFimex::Data::asConstDouble () const [pure virtual]

retrieve data as double

15.33.3.4 virtual boost::shared_array<const float> MetNoFimex::Data::asConstFloat () const [pure virtual]

retrieve data as float

15.33.3.5 virtual boost::shared_array<const int> MetNoFimex::Data::asConstInt () const [pure virtual]

retrieve data as int

15.33.3.6 virtual boost::shared_array<const long long> MetNoFimex::Data::asConstInt64 () const [pure virtual]

retrieve data as int64

15.33.3.7 virtual boost::shared_array<const short> MetNoFimex::Data::asConstShort () const [pure virtual]

retrieve data as short

15.33.3.8 virtual boost::shared_array<const unsigned char> MetNoFimex::Data::asConstUChar () const [pure virtual]

retrieve data as uchar

15.33.3.9 virtual boost::shared_array<const unsigned int> MetNoFimex::Data::asConstUInt () const [pure virtual]

retrieve data as uint

15.33.3.10 virtual boost::shared_array<const unsigned long long> MetNoFimex::Data::asConstUInt64 () const [pure virtual]

retrieve data as uint64

15.33.3.11 virtual boost::shared_array<const unsigned short> MetNoFimex::Data::asConstUShort () const [pure virtual]

retrieve data as short

15.33.3.12 virtual boost::shared_array<double> MetNoFimex::Data::asDouble () [pure virtual]

retrieve data as double

15.33.3.13 virtual boost::shared_array<float> MetNoFimex::Data::asFloat () [pure virtual]

retrieve data as float (eventually copy)

15.33.3.14 virtual boost::shared_array<int> MetNoFimex::Data::asInt () [pure virtual]

retrieve data as int

15.33.3.15 virtual boost::shared_array<long long> MetNoFimex::Data::asInt64 () [pure virtual]

retrieve data as int64

15.33.3.16 virtual boost::shared_array<short> MetNoFimex::Data::asShort () [pure virtual]

retrieve data as short

15.33.3.17 virtual std::string MetNoFimex::Data::asString (std::string *separator* = "") const [pure virtual]

retrieve the whole array as a string (with possible separator)

15.33.3.18 virtual boost::shared_array<unsigned char> MetNoFimex::Data::asUChar () [pure virtual]

retrieve data as uchar

15.33.3.19 virtual boost::shared_array<unsigned int> MetNoFimex::Data::asUInt () [pure virtual]

retrieve data as uint

15.33.3.20 virtual boost::shared_array<unsigned long long> MetNoFimex::Data::asUInt64 () [pure virtual]

retrieve data as uint64

15.33.3.21 virtual boost::shared_array<unsigned short> MetNoFimex::Data::asUShort () [pure virtual]

retrieve data as short

15.33.3.22 virtual int MetNoFimex::Data::bytes_for_one () const [pure virtual]

sizeof the data-impl datatype

15.33.3.23 virtual boost::shared_ptr<Data> MetNoFimex::Data::clone () const [pure virtual]

duplicate the data

The clone operation generates a real duplicate of the data. The internal array-data will be copied.

15.33.3.24 virtual boost::shared_ptr<Data> MetNoFimex::Data::convertDataType (double *oldFill*, double *oldScale*, double *oldOffset*, CDMDataType *newType*, double *newFill*, double *newScale*, double *newOffset*) [pure virtual]

convert the datatype from one type,fill,scale,offset to another

15.33.3.25 virtual void* MetNoFimex::Data::getDataPtr () [pure virtual]

15.33.3.26 virtual CDMDataType MetNoFimex::Data::getDataType () const [pure virtual]

return the CDMDataType of this data

15.33.3.27 virtual void MetNoFimex::Data::setAllValues (double *val*) [pure virtual]

set all values to the submitted value

Parameters

val value to set

15.33.3.28 virtual void MetNoFimex::Data::setValue (long *pos*, double *val*) [pure virtual]

set a value at the desired position

15.33.3.29 virtual void MetNoFimex::Data::setValues (size_t *startPos*, const Data & *data*, size_t *first* = 0, size_t *end* = -1) [pure virtual]

set the values from another Data implementation

Parameters

startPos the first position the data should be written to

data the other data-source

first the first data-entry

end the last (excluded) data-entry, defaults to MAX size_t, automatically shrunken to fit size

15.33.3.30 virtual size_t MetNoFimex::Data::size () const [pure virtual]

size of the data

15.33.3.31 virtual boost::shared_ptr<Data> MetNoFimex::Data::slice (std::vector< size_t > *orgDimSize*, std::vector< size_t > *startDims*, std::vector< size_t > *outputDimSize*) [pure virtual]

get a multi-dimensional slice of the data

This slices a multidimensional chunk out of the data. All parameters must be vectors of the same size (dimension of array). The first dimension is the fastest moving index (fortran arrays)

Parameters

orgDimSize the dimensions of this vector. The product of all orgDimSizes must equal to data.size.

startDims The start-position in the original data to fetch data from

outputDimSize the size of the output data

Returns

a Data of the size of outputDimSize with the same datatype as the original type

Exceptions

CDMException on dimension mismatch: (start+size > orgDimSize) or (Product(orgDimSize) != size)

15.33.3.32 virtual void MetNoFimex::Data::toStream (std::ostream &, std::string *separator* = "") const [pure virtual]

printing of the current data to ostream, with optional separator

The documentation for this class was generated from the following file:

- include/fimex/Data.h

15.34 MetNoFimex::DataTypeChanger Class Reference

`#include <DataTypeChanger.h>`

Public Member Functions

- DataTypeChanger (CDMDataType oldType)
- DataTypeChanger (CDMDataType oldType, double oldFill, double oldScale, double oldOffset, CDMDataType newType, double newFill, double newScale, double newOffset, double unitScale=1., double unitOffset=0.)
- virtual ~DataTypeChanger ()
- boost::shared_ptr< Data > convertData (boost::shared_ptr< Data >) const throw (CDMException)
- CDMDataType getDataType () const

15.34.1 Detailed Description

brief wrapper class around data->convertType

15.34.2 Constructor & Destructor Documentation

15.34.2.1 MetNoFimex::DataTypeChanger::DataTypeChanger (CDMDataType *oldType*) [explicit]

initialize data with the oldType convertData will do nothing in this case

15.34.2.2 MetNoFimex::DataTypeChanger::DataTypeChanger (CDMDataType *oldType*, double *oldFill*, double *oldScale*, double *oldOffset*, CDMDataType *newType*, double *newFill*, double *newScale*, double *newOffset*, double *unitScale* = 1., double *unitOffset* = 0.) [explicit]

initialize with the old and new settings

Parameters

oldType datatype of original data

oldFill fill value of the original data

oldScale scale_factor of the original data

oldOffset scale_factor of the original data

newType datatype of converted data

newFill fill value of converted data

newScale scale_factor of the converted data

newOffset add_offset of the converted data

unitScale scale_factor for the unpacked data, i.e. for unit changes, default 1.

unitOffset offset for the unpacked data, i.e. for unit changes, default 0.

15.34.2.3 virtual MetNoFimex::DataTypeChanger::~DataTypeChanger () [virtual]

15.34.3 Member Function Documentation

15.34.3.1 boost::shared_ptr<Data> MetNoFimex::DataTypeChanger::convertData (boost::shared_ptr< Data >) const throw (CDMException)

convert the data to the new scale/fill/offset

15.34.3.2 CDMDataType MetNoFimex::DataTypeChanger::getDataType () const

return the datatype of the converted data

The documentation for this class was generated from the following file:

- include/fimex/DataTypeChanger.h

15.35 MetNoFelt::Felt_Array2 Class Reference

A collection of FeltFields to build a 4-dimensional data array for 1-parameter and 1-vertical coordinate.

`#include <Felt_Array2.h>`

Public Member Functions

- Felt_Array2 (const string name, const boost::shared_ptr< felt::FeltField > feltField, const string &dataType, double fillValue)
- virtual ~Felt_Array2 ()
- void addInformationByField (boost::shared_ptr< felt::FeltField > field)
- const string & getName () const

  get the time/level independent data-header

- const string & getDatatype () const
- int getGrid (boost::posix_time::ptime time, **LevelPair** levelPair, **vector**< short > &gridOut)
- int getGridAllowDelta (boost::posix_time::ptime time, **LevelPair** levelPair, **vector**< short > &gridOut, const boost::array< float, 6 > &gridParameterDelta)
- int getLevelType () const

  get the felt level type of this array

- double getFillValue () const
- **vector**< boost::posix_time::ptime > getTimes () const
- **vector**< boost::posix_time::ptime > getReferenceTimes () const
- **vector**< **LevelPair** > getLevelPairs () const
- **vector**< short > getEnsembleMembers () const
- int getIdent19 (boost::posix_time::ptime time, **LevelPair** levelPair) const
- int getX () const
- int getY () const
- int scaleFactor () const
- double getScalingFactor () const
- boost::shared_ptr< felt::FeltGridDefinition > getGridDefinition () const
- int getGridType () const
- bool hasTime () const
- const boost::shared_ptr< felt::FeltField > getField (boost::posix_time::ptime time, **LevelPair** levelPair) const

15.35.1 Detailed Description

A collection of FeltFields to build a 4-dimensional data array for 1-parameter and 1-vertical coordinate. A Felt_Array2 collects all felt::FeltField from a felt::FeltFile with the same parameter and vertical coordinate. It is possible to access the data of such each layer.

15.35.2 Constructor & Destructor Documentation

15.35.2.1 MetNoFelt::Felt_Array2::Felt_Array2 (const string *name*, const boost::shared_ptr< felt::FeltField > *feltField*, const string & *dataType*, double *fillValue*) [explicit]

constructor applying the parameter name and the felt description from the first feltField

Parameters

name parameter name

field a field added to this constructor

dataType short|float|double datatype used for autoscaling, getScalingFactor() will be always return 1 for float and double

fillValue fillValue of the datatype, usually -32767

15.35.2.2 virtual MetNoFelt::Felt_Array2::~Felt_Array2 () [virtual]

15.35.3 Member Function Documentation

15.35.3.1 void MetNoFelt::Felt_Array2::addInformationByField (boost::shared_ptr< felt::FeltField > *field*)

add information from the felt-index (usually retrieved from qfelt) to this Felt_Array2 the index given here must correspond to the initialization index

15.35.3.2 const string& MetNoFelt::Felt_Array2::getDatatype () const [inline]

return the datatype as string short|float|double

15.35.3.3 vector<short> MetNoFelt::Felt_Array2::getEnsembleMembers () const

Returns

the ensemble member

15.35.3.4 const boost::shared_ptr<felt::FeltField> MetNoFelt::Felt_Array2::getField (boost::posix_time::ptime *time*, LevelPair *levelPair*) const

15.35.3.5 double MetNoFelt::Felt_Array2::getFillValue () const [inline]

return the changed fill used in Felt_File::getScaledDataSlice

15.35.3.6 int MetNoFelt::Felt_Array2::getGrid (boost::posix_time::ptime *time*, LevelPair *levelPair*, vector< short > & *gridOut*)

read a grid for a time and a levelPair

Parameters

time The time of the field

levelPair The levelPair of the field

gridOut The data of this field will be put into this grid

Returns

the scaleFactor as tenth exponent of this field (grid * 10^scaleFactor)

Exceptions

Felt_File_Error if the gridDefinition (gridType or gridParameters) change

15.35.3.7 int MetNoFelt::Felt_Array2::getGridAllowDelta (boost::posix_time::ptime *time*, LevelPair *levelPair*, vector< short > & *gridOut*, const boost::array< float, 6 > & *gridParameterDelta*)

same as getGrid, but the gridParameters to change up to the value provided in gridParameterDelta

15.35.3.8 boost::shared_ptr<felt::FeltGridDefinition> MetNoFelt::Felt_Array2::getGridDefinition () const

15.35.3.9 int MetNoFelt::Felt_Array2::getGridType () const

15.35.3.10 int MetNoFelt::Felt_Array2::getIdent19 (boost::posix_time::ptime *time*, LevelPair *levelPair*) const

get the ident19 parameter from the data-header, throw error if levelPair/time doesn't exists

Warning

only ident19 of data already read will be taken into account

15.35.3.11 vector<LevelPair> MetNoFelt::Felt_Array2::getLevelPairs () const

return the level pairs (niveau 1, niveau 2) for this parameter as used by hybrid levels for ensemble, niveau 2 is set to 0 and should be retrieved from the ensemble-members

15.35.3.12 int MetNoFelt::Felt_Array2::getLevelType () const

get the felt level type of this array

15.35.3.13 const string& MetNoFelt::Felt_Array2::getName () const

get the time/level independent data-header

return the parameter name

15.35.3.14 vector<boost::posix_time::ptime> MetNoFelt::Felt_Array2::getReferenceTimes () const

return the reference-times for this parameter, sorted by getTimes()

15.35.3.15 double MetNoFelt::Felt_Array2::getScalingFactor () const

Returns

scalingFactor

15.35.3.16 vector<boost::posix_time::ptime> MetNoFelt::Felt_Array2::getTimes () const

return the times available for this parameter, sorted

15.35.3.17 int MetNoFelt::Felt_Array2::getX () const

Returns

x/longitude size

15.35.3.18 int MetNoFelt::Felt_Array2::getY () const

Returns

y/latitude size

15.35.3.19 bool MetNoFelt::Felt_Array2::hasTime () const

Returns

true if grid has a time-axis, i.e. not a parameter field

15.35.3.20 int MetNoFelt::Felt_Array2::scaleFactor () const

get the files scaleFactor, this corresponds to scalingFactor by 10^(scaleFactor) == scalingFactor

The documentation for this class was generated from the following file:

- include/fimex/Felt_Array2.h

15.36 MetNoFelt::Felt_File2 Class Reference

Felt File access.

`#include <Felt_File2.h>`

Public Member Functions

- Felt_File2 ()
  constructor
- Felt_File2 (const **std::string** &filename)
- Felt_File2 (const **std::string** &filename, const **std::vector**< **std::string** > &dianaParamList, const **std::map**< **std::string**, **std::string** > &options)
- virtual ~Felt_File2 ()
- const boost::shared_ptr< Felt_Array2 > getFeltArray (const **std::string** &compName) const
  retrieve a Felt_Array2
- boost::shared_ptr< MetNoFimex::Data > getScaledDataSlice (boost::shared_ptr< Felt_Array2 > feltArray, const boost::posix_time::ptime time, const **LevelPair** level)
  retrieve a data slice
- **std::vector**< boost::shared_ptr< Felt_Array2 > > listFeltArrays () const
- **std::map**< short, **std::vector**< **LevelPair** > > getFeltLevelPairs () const
- **std::vector**< short > getEnsembleMembers () const
- const **std::map**< **LevelPair**, int > & getHybridLevels () const
- **std::vector**< boost::posix_time::ptime > getFeltTimes () const
  all time values, sorted
- boost::shared_ptr< boost::posix_time::ptime > getUniqueReferenceTime () const
- int getNX () const
  get size in x direction
- int getNY () const
  get size in y direction
- boost::shared_ptr< MetNoFimex::Data > getXData () const
  get the values of the x axis
- boost::shared_ptr< MetNoFimex::Data > getYData () const
  get the values of the y axis
- int getGridType () const
- boost::shared_ptr< felt::FeltGridDefinition > getGridDefinition () const
  assumes one set of grid-parameters for the whole file

15.36.1 Detailed Description

Felt File access. Felt_File2 gives c++ style access to felt files. It uses internally libmi and caches the table of contents

15.36.2 Constructor & Destructor Documentation

15.36.2.1 MetNoFelt::Felt_File2::Felt_File2 () [inline]

constructor

open an empty felt file, just a default constructor, no useful information

15.36.2.2 MetNoFelt::Felt_File2::Felt_File2 (const std::string & *filename*) [explicit]

open and read toc of a felt file

Parameters

filename name of felt file

15.36.2.3 MetNoFelt::Felt_File2::Felt_File2 (const std::string & *filename*, const std::vector< std::string > & *dianaParamList*, const std::map< std::string, std::string > & *options*) [explicit]

open and read toc of a felt file

Parameters

paramList a list of known parameters (in diana format, e.g. 17,2,1000:prod=74), only the known parameters will be read

Warning

The diana format is extended by dataType=short|float|double and fillValue=(number in short|float|double) to add the return type of the data. Autoscaling will be turned on for 'getDataSlice'. default is dataType=short:fillValue=-32767

15.36.2.4 virtual MetNoFelt::Felt_File2::~Felt_File2 () [virtual]

15.36.3 Member Function Documentation

15.36.3.1 std::vector<short> MetNoFelt::Felt_File2::getEnsembleMembers () const

get all members of ensembles

15.36.3.2 const boost::shared_ptr<Felt_Array2> MetNoFelt::Felt_File2::getFeltArray (const std::string & *compName*) const

retrieve a Felt_Array2

Parameters

compName parameter name of felt file as named in diana setup

15.36.3.3 std::map<short, std::vector<LevelPair> > MetNoFelt::Felt_File2::getFeltLevelPairs () const

Z-axis types and values

Returns

map consisting of felt level-ids and a sorted vector of level values Z-axis types and values
map consisting of felt level-ids and a sorted vector of level-pairs of values

15.36.3.4 std::vector<boost::posix_time::ptime> MetNoFelt::Felt_File2::getFeltTimes () const

all time values, sorted

15.36.3.5 boost::shared_ptr<felt::FeltGridDefinition> MetNoFelt::Felt_File2::getGridDefinition () const

assumes one set of grid-parameters for the whole file

15.36.3.6 int MetNoFelt::Felt_File2::getGridType () const

assumes one set of grid-parameters for the whole file, returns parameter between 1 and 6, without extra definition

15.36.3.7 const std::map<LevelPair, int>& MetNoFelt::Felt_File2::getHybridLevels () const [inline]

15.36.3.8 int MetNoFelt::Felt_File2::getNX () const

get size in x direction

15.36.3.9 int MetNoFelt::Felt_File2::getNY () const

get size in y direction

15.36.3.10 boost::shared_ptr<MetNoFimex::Data> MetNoFelt::Felt_File2::getScaledDataSlice (boost::shared_ptr< Felt_Array2 > *feltArray*, const boost::posix_time::ptime *time*, const LevelPair *level*)

retrieve a data slice

retrieve the data prescaled (if float or double) and replaced with the new fill value

Parameters

compName parameter name of felt file

time time of slice

level level of slice

15.36.3.11 boost::shared_ptr<boost::posix_time::ptime> MetNoFelt::Felt_File2::getUniqueReferenceTime () const

get the unique reference time of the felt file

Returns

a unique reference time

Exceptions

exception if no unique reference time exists

15.36.3.12 boost::shared_ptr<MetNoFimex::Data> MetNoFelt::Felt_File2::getXData () const

get the values of the x axis

15.36.3.13 boost::shared_ptr<MetNoFimex::Data> MetNoFelt::Felt_File2::getYData () const

get the values of the y axis

15.36.3.14 std::vector<boost::shared_ptr<Felt_Array2> > MetNoFelt::Felt_File2::listFeltArrays () const

retrieve all felt arrays

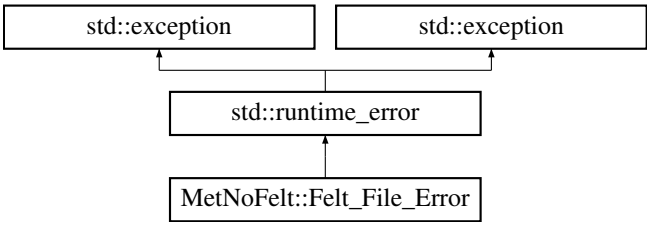
The documentation for this class was generated from the following file:

- include/fimex/Felt_File2.h

15.37 MetNoFelt::Felt_File_Error Class Reference

`#include <Felt_File_Error.h>`

Inheritance diagram for MetNoFelt::Felt_File_Error:



Public Member Functions

- Felt_File_Error (const **std::string** &message)

15.37.1 Constructor & Destructor Documentation

15.37.1.1 MetNoFelt::Felt_File_Error::Felt_File_Error (const std::string & *message*) [inline, explicit]

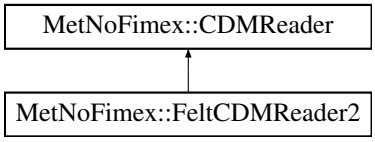
The documentation for this class was generated from the following file:

- include/fimex/Felt_File_Error.h

15.38 MetNoFimex::FeltCDMReader2 Class Reference

`#include <FeltCDMReader2.h>`

Inheritance diagram for MetNoFimex::FeltCDMReader2:



Public Member Functions

- FeltCDMReader2 (**std::string** filename, const XMLInput &configInput)
- FeltCDMReader2 (**std::string** filename, **std::string** configFilename)
- virtual ~FeltCDMReader2 ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos)

  data-reading function to be called from the CDMWriter

15.38.1 Constructor & Destructor Documentation

15.38.1.1 MetNoFimex::FeltCDMReader2::FeltCDMReader2 (std::string *filename*, const XMLInput & *configInput*)

15.38.1.2 MetNoFimex::FeltCDMReader2::FeltCDMReader2 (std::string *filename*, std::string *configFilename*)

15.38.1.3 virtual MetNoFimex::FeltCDMReader2::~FeltCDMReader2 () `[virtual]`

15.38.2 Member Function Documentation

15.38.2.1 virtual boost::shared_ptr<Data> MetNoFimex::FeltCDMReader2::getDataSlice (const std::string & *varName*, size_t *unLimDimPos*) `[virtual]`

data-reading function to be called from the CDMWriter

This methods needs to be implemented by the CDMReader. It should provide the data for each variable, either by reading from disk, converting from another CDMReader or reading from an in-memory data-section.

This function should retrieve the whole data for a dataset without unlimited dimension if the unLimDimPos == 0.

Parameters

varName name of the variable to read

unLimDimPos (optional) if the variable contains a unlimited dimension (max one allowed) an slice of this position is returned

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

Implements MetNoFimex::CDMReader.

The documentation for this class was generated from the following file:

- include/fimex/FeltCDMReader2.h

15.39 felt::FeltField Class Reference

`#include <FeltField.h>`

Public Types

- typedef boost::array< word, 16 > Header

Public Member Functions

- FeltField (const FeltFile &ff, size_t index)
- ~FeltField ()
- bool valid () const
- int producer () const
- int gridArea () const
- boost::posix_time::ptime referenceTime () const
- boost::posix_time::ptime validTime () const
- int parameter () const
- int dataType () const
- int verticalCoordinate () const
- int level1 () const
- int level2 () const
- int gridType () const
- void grid (**std::vector**< word > &out) const
- size_t gridSize () const
- int scaleFactor () const
- int xNum () const
- int yNum () const
- int miscField () const

  this field is described in the felt documentation as "word 19 in data part"

- bool isEpsRunParameter () const
- int dataVersion () const
- FeltGridDefinitionPtr projectionInformation () const

  throws

- **std::string** information () const
- **std::string** gridInformation () const
- const Header & getHeader () const

  access felt index header

15.39.1 Member Typedef Documentation

15.39.1.1 typedef boost::array<word, 16> felt::FeltField::Header

15.39.2 Constructor & Destructor Documentation

15.39.2.1 felt::FeltField::FeltField (const FeltFile & *ff*, size_t *index*)

15.39.2.2 felt::FeltField::~FeltField ()

15.39.3 Member Function Documentation

15.39.3.1 int felt::FeltField::dataType () const [inline]

read the time dataType, i.e. 1=analysis 2=interpolated/initialization 3=prognosis 4=parameter-field(no time)

15.39.3.2 int felt::FeltField::dataVersion () const

Get data version if this is an eps parameter, otherwise 0

15.39.3.3 const Header& felt::FeltField::getHeader () const [inline]

access felt index header

15.39.3.4 void felt::FeltField::grid (std::vector< word > & *out*) const

Read the grid from file.

15.39.3.5 int felt::FeltField::gridArea () const [inline]

15.39.3.6 std::string felt::FeltField::gridInformation () const

15.39.3.7 size_t felt::FeltField::gridSize () const

15.39.3.8 int felt::FeltField::gridType () const [inline]

15.39.3.9 std::string felt::FeltField::information () const

15.39.3.10 bool felt::FeltField::isEpsRunParameter () const [inline]

15.39.3.11 int felt::FeltField::level1 () const

Get primary level value

15.39.3.12 int felt::FeltField::level2 () const

Get the secondary level value, or 0 if that level field has internally been used for something else. This will happen if the field is part of an enseble run.

15.39.3.13 int felt::FeltField::miscField () const

this field is described in the felt documentation as "word 19 in data part"

15.39.3.14 int felt::FeltField::parameter () const

Get the parameter value. This is the logical value, which means that if the parameter is part of an ensemble run, it will not start with 2000,3000,4000.

15.39.3.15 int felt::FeltField::producer () const [inline]

15.39.3.16 FeltGridDefinitionPtr felt::FeltField::projectionInformation () const

throws

15.39.3.17 boost::posix_time::ptime felt::FeltField::referenceTime () const

15.39.3.18 int felt::FeltField::scaleFactor () const

15.39.3.19 bool felt::FeltField::valid () const [inline]

15.39.3.20 boost::posix_time::ptime felt::FeltField::validTime () const

15.39.3.21 int felt::FeltField::verticalCoordinate () const [inline]

15.39.3.22 int felt::FeltField::xNum () const

15.39.3.23 int felt::FeltField::yNum () const

The documentation for this class was generated from the following file:

- include/felt/FeltField.h

15.40 felt::FeltFile Class Reference

`#include <FeltFile.h>`

Public Types

- typedef size_t size_type
- typedef boost::shared_ptr< FeltField > FeltFieldPtr
- typedef **std::vector**< FeltFieldPtr >::const_iterator iterator
- typedef iterator const_iterator

Public Member Functions

- FeltFile (const boost::filesystem::path &file)
- ~FeltFile ()
- size_type size () const
- bool empty () const
- const boost::filesystem::path & fileName () const
- **std::string** information () const
- boost::posix_time::ptime lastUpdateTime () const
- boost::posix_time::ptime referenceTime () const
- boost::posix_time::ptime firstTime () const
- boost::posix_time::ptime lastTime () const
- iterator begin ()
- iterator end ()
- const_iterator begin () const
- const_iterator end () const
- const FeltField & at (size_t idx) const

  throws **std::out_of_range** *if idx is too large.*

Static Public Member Functions

- static void log (const **std::string** &msg)
- static void setLogStream (**std::ostream** &o)
- static void setLogging (bool enableLogging)
- static bool isLogging ()

Friends

- class FeltField

15.40.1 Member Typedef Documentation

15.40.1.1 typedef iterator felt::FeltFile::const_iterator

15.40.1.2 typedef boost::shared_ptr<FeltField> felt::FeltFile::FeltFieldPtr

15.40.1.3 typedef std::vector<FeltFieldPtr>::const_iterator felt::FeltFile::iterator

15.40.1.4 typedef size_t felt::FeltFile::size_type

15.40.2 Constructor & Destructor Documentation

15.40.2.1 felt::FeltFile::FeltFile (const boost::filesystem::path & *file*) `[explicit]`

15.40.2.2 felt::FeltFile::~FeltFile ()

15.40.3 Member Function Documentation

15.40.3.1 const FeltField& felt::FeltFile::at (size_t *idx*) const

throws **std::out_of_range** if idx is too large.

15.40.3.2 const_iterator felt::FeltFile::begin () const

15.40.3.3 iterator felt::FeltFile::begin ()

15.40.3.4 bool felt::FeltFile::empty () const [inline]

15.40.3.5 const_iterator felt::FeltFile::end () const

15.40.3.6 iterator felt::FeltFile::end ()

15.40.3.7 const boost::filesystem::path& felt::FeltFile::fileName () const [inline]

15.40.3.8 boost::posix_time::ptime felt::FeltFile::firstTime () const

15.40.3.9 std::string felt::FeltFile::information () const

15.40.3.10 static bool felt::FeltFile::isLogging () [static]

15.40.3.11 boost::posix_time::ptime felt::FeltFile::lastTime () const

15.40.3.12 boost::posix_time::ptime felt::FeltFile::lastUpdateTime () const

15.40.3.13 static void felt::FeltFile::log (const std::string & *msg*) [static]

15.40.3.14 boost::posix_time::ptime felt::FeltFile::referenceTime () const

15.40.3.15 static void felt::FeltFile::setLogging (bool *enableLogging*) [static]

15.40.3.16 static void felt::FeltFile::setLogStream (std::ostream & *o*) [static]

15.40.3.17 size_type felt::FeltFile::size () const

15.40.4 Friends And Related Function Documentation

15.40.4.1 friend class FeltField [friend]

The documentation for this class was generated from the following file:

- include/felt/FeltFile.h

15.41 felt::FeltGridDefinition Class Reference

`#include <FeltGridDefinition.h>`

Public Types

- enum Orientation { LeftUpperHorizontal = 0, LeftLowerHorizontal = 64 }

Public Member Functions

- FeltGridDefinition (int gridType, int xNum, int yNum, int a, int b, int c, int d, const **std::vector**< short int > &extraData)
- virtual ~FeltGridDefinition ()
- virtual **std::string** projDefinition () const
- virtual int getXNumber () const
- virtual int getYNumber () const
- virtual float getXIncrement () const
- virtual float getYIncrement () const
- virtual float startLongitude () const
- virtual float startLatitude () const
- virtual float startX () const
- virtual float startY () const
- virtual const boost::array< float, 6 > & getGridParameters () const
- Orientation getScanMode () const

15.41.1 Member Enumeration Documentation

15.41.1.1 enum felt::FeltGridDefinition::Orientation

Orientation describes the different ways that the values can be ordered in the grid. There are four possible dimensions: Left to Right or Right to Left Lower to Upper or Upper to Lower Horizontal scanning or Vertical scanning Regular or Alternating (i.e., every second row changes direction)

Enumerator:

LeftUpperHorizontal

LeftLowerHorizontal

15.41.2 Constructor & Destructor Documentation

15.41.2.1 felt::FeltGridDefinition::FeltGridDefinition (int *gridType*, int *xNum*, int *yNum*, int *a*, int *b*, int *c*, int *d*, const std::vector< short int > & *extraData*)

The parameters a, b, c, d are words 15 to 18 in the FELT header definition. These usually describe elements of the grid specification (variable meaning, depending on the grid specification used)

15.41.2.2 virtual felt::FeltGridDefinition::~FeltGridDefinition () [virtual]

15.41.3 Member Function Documentation

15.41.3.1 virtual const boost::array<float, 6>& felt::FeltGridDefinition::getGridParameters () const [virtual]

15.41.3.2 Orientation felt::FeltGridDefinition::getScanMode () const

15.41.3.3 virtual float felt::FeltGridDefinition::getXIncrement () const [virtual]

15.41.3.4 virtual int felt::FeltGridDefinition::getXNumber () const [virtual]

15.41.3.5 virtual float felt::FeltGridDefinition::getYIncrement () const [virtual]

15.41.3.6 virtual int felt::FeltGridDefinition::getYNumber () const [virtual]

15.41.3.7 virtual std::string felt::FeltGridDefinition::projDefinition () const [virtual]

15.41.3.8 virtual float felt::FeltGridDefinition::startLatitude () const [virtual]

15.41.3.9 virtual float felt::FeltGridDefinition::startLongitude () const [virtual]

15.41.3.10 virtual float felt::FeltGridDefinition::startX () const [virtual]

15.41.3.11 virtual float felt::FeltGridDefinition::startY () const [virtual]

The documentation for this class was generated from the following file:

- include/felt/FeltGridDefinition.h

15.42 MetNoFelt::FeltParameters Class Reference

`#include <FeltParameters.h>`

Public Member Functions

- FeltParameters ()
- FeltParameters (**std::string** filename)
- FeltParameters (const **std::vector**< **std::string** > &feltParams, const **std::string** &globalRestrictions)
- virtual ~FeltParameters ()
- const boost::array< short, 16 > & getParameters (const **std::string** &)
- const **std::string** & getParameterName (const boost::array< short, 16 > &)
- **std::string** getParameterDatatype (const **std::string** &parameterName) const
- double getParameterFillValue (const **std::string** &parameterName) const

Static Public Member Functions

- static const **std::string** & DEFAULT_CONFIG ()

15.42.1 Constructor & Destructor Documentation

15.42.1.1 MetNoFelt::FeltParameters::FeltParameters ()

15.42.1.2 MetNoFelt::FeltParameters::FeltParameters (std::string *filename*) `[explicit]`

initialize all known felt parameters from a diana-setup file

Parameters

filename diana setup file

15.42.1.3 MetNoFelt::FeltParameters::FeltParameters (const std::vector< std::string > & *feltParams*, const std::string & *globalRestrictions*) `[explicit]`

initialize parameters from a list of parameters in diana format, e.g. 17,2,1000:prod=74

15.42.1.4 virtual MetNoFelt::FeltParameters::~FeltParameters () [virtual]

15.42.2 Member Function Documentation

15.42.2.1 static const std::string& MetNoFelt::FeltParameters::DEFAULT_CONFIG () [inline, static]

15.42.2.2 std::string MetNoFelt::FeltParameters::getParameterDatatype (const std::string & *parameterName*) const

15.42.2.3 double MetNoFelt::FeltParameters::getParameterFillValue (const std::string & *parameterName*) const

15.42.2.4 const std::string& MetNoFelt::FeltParameters::getParameterName (const boost::array< short, 16 > &)

15.42.2.5 const boost::array<short, 16>& MetNoFelt::FeltParameters::getParameters (const std::string &)

The documentation for this class was generated from the following file:

- include/fimex/FeltParameters.h

15.43 MetNoFimex::FimexTime Class Reference

`#include <TimeUnit.h>`

Public Types

- enum special_values { min_date_time, max_date_time }

Public Member Functions

- FimexTime ()
- FimexTime (unsigned short year, char month, char mday, char hour=0, char minute=0, char second=0, unsigned short msecond=0)
- FimexTime (special_values val)
- bool parseISO8601 (const **std::string** &isoString)
- void setTime (unsigned short year, char month, char mday, char hour=0, char minute=0, char second=0, unsigned short msecond=0)

  set all the time-parameters at once

- unsigned short getYear () const

  year (2008 as of writing)

- void setYear (unsigned short year)
- char getMonth () const

  month (1-12)

- void setMonth (char month)
- char getMDay () const

  day of month (1-31)

- void setMDay (char mday)
- char getHour () const

  hour (0-23)

- void setHour (char hour)
- char getMinute () const

  minute (0-59)

- void setMinute (char minute)
- char getSecond () const

  second (0-59)

- void setSecond (char second)
- unsigned short getMSecond () const

  millisecond

- void setMSecond (unsigned short msecond)
- bool operator== (const FimexTime &rhs) const

  compare two fimexTimes

- bool operator!= (const FimexTime &rhs) const
  compare two fimexTimes
- bool operator> (const FimexTime &rhs) const
  compare two fimexTimes
- bool operator< (const FimexTime &rhs) const
  compare two fimexTimes
- bool operator>= (const FimexTime &rhs) const
  compare two fimexTimes
- bool operator<= (const FimexTime &rhs) const
  compare two fimexTimes

15.43.1 Detailed Description

time representation and some overloaded operators

Warning

: the implementor needs to make sure, that all values are given correctly, i.e. seconds between 0 and 59

15.43.2 Member Enumeration Documentation

15.43.2.1 enum MetNoFimex::FimexTime::special_values

Enumerator:

min_date_time

max_date_time

15.43.3 Constructor & Destructor Documentation

15.43.3.1 MetNoFimex::FimexTime::FimexTime () [inline]

15.43.3.2 MetNoFimex::FimexTime::FimexTime (unsigned short *year*, char *month*, char *mday*, char *hour* = 0, char *minute* = 0, char *second* = 0, unsigned short *msecond* = 0)

15.43.3.3 MetNoFimex::FimexTime::FimexTime (special_values *val*)

15.43.4 Member Function Documentation

15.43.4.1 char MetNoFimex::FimexTime::getHour () const [inline]

hour (0-23)

15.43.4.2 char MetNoFimex::FimexTime::getMDay () const [inline]

day of month (1-31)

15.43.4.3 char MetNoFimex::FimexTime::getMinute () const [inline]

minute (0-59)

15.43.4.4 char MetNoFimex::FimexTime::getMonth () const [inline]

month (1-12)

15.43.4.5 unsigned short MetNoFimex::FimexTime::getMSecond () const [inline]

millisecond

15.43.4.6 char MetNoFimex::FimexTime::getSecond () const [inline]

second (0-59)

15.43.4.7 unsigned short MetNoFimex::FimexTime::getYear () const [inline]

year (2008 as of writing)

15.43.4.8 bool MetNoFimex::FimexTime::operator!= (const FimexTime & *rhs*) const [inline]

compare two fimexTimes

15.43.4.9 bool MetNoFimex::FimexTime::operator< (const FimexTime & *rhs*) const [inline]

compare two fimexTimes

15.43.4.10 bool MetNoFimex::FimexTime::operator<= (const FimexTime & *rhs*) const [inline]

compare two fimexTimes

15.43.4.11 bool MetNoFimex::FimexTime::operator== (const FimexTime & *rhs*) const

compare two fimexTimes

15.43.4.12 bool MetNoFimex::FimexTime::operator> (const FimexTime & *rhs*) const [inline]

compare two fimexTimes

15.43.4.13 bool MetNoFimex::FimexTime::operator>= (const FimexTime & *rhs*) const [inline]

compare two fimexTimes

15.43.4.14 bool MetNoFimex::FimexTime::parseISO8601 (const std::string & *isoString*)

parse and set the time in ISO8601 formats (not all), e.g. YYYY-MM-DD, HH:MM:SS, YYYY-MM-DD HH:MM:SS, YYYY-MM-DDTHH:MM:SS (and without seconds)

Returns

true, if time/date has been set, false otherwise

15.43.4.15 void MetNoFimex::FimexTime::setHour (char *hour*) [inline]

15.43.4.16 void MetNoFimex::FimexTime::setMDay (char *mday*) [inline]

15.43.4.17 void MetNoFimex::FimexTime::setMinute (char *minute*) [inline]

15.43.4.18 void MetNoFimex::FimexTime::setMonth (char *month*) [inline]

15.43.4.19 void MetNoFimex::FimexTime::setMSecond (unsigned short *msecond*) [inline]

15.43.4.20 void MetNoFimex::FimexTime::setSecond (char *second*) [inline]

15.43.4.21 void MetNoFimex::FimexTime::setTime (unsigned short *year*, char *month*, char *mday*, char *hour* = 0, char *minute* = 0, char *second* = 0, unsigned short *msecond* = 0)

set all the time-parameters at once

15.43.4.22 void MetNoFimex::FimexTime::setYear (unsigned short *year*) [inline]

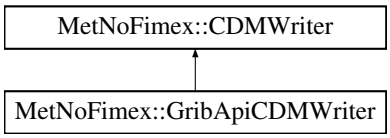
The documentation for this class was generated from the following file:

- include/fimex/TimeUnit.h

15.44 MetNoFimex::GribApiCDMWriter Class Reference

`#include <GribApiCDMWriter.h>`

Inheritance diagram for MetNoFimex::GribApiCDMWriter:



Public Member Functions

- GribApiCDMWriter (const boost::shared_ptr< CDMReader > cdmReader, const **std::string** &outputFile, const int gribVersion, const **std::string** &configFile)
- virtual ~GribApiCDMWriter ()

15.44.1 Constructor & Destructor Documentation

15.44.1.1 MetNoFimex::GribApiCDMWriter::GribApiCDMWriter (const boost::shared_ptr< CDMReader > *cdmReader*, const std::string & *outputFile*, const int *gribVersion*, const std::string & *configFile*)

15.44.1.2 virtual MetNoFimex::GribApiCDMWriter::~GribApiCDMWriter () `[virtual]`

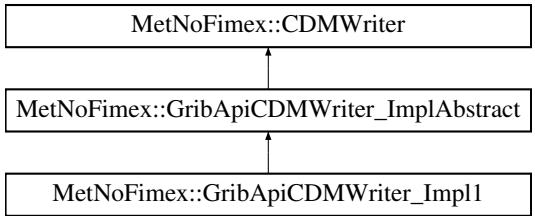
The documentation for this class was generated from the following file:

- include/fimex/GribApiCDMWriter.h

15.45 MetNoFimex::GribApiCDMWriter_Impl1 Class Reference

`#include <GribApiCDMWriter_Impl1.h>`

Inheritance diagram for MetNoFimex::GribApiCDMWriter_Impl1:



Public Member Functions

- GribApiCDMWriter_Impl1 (const boost::shared_ptr< CDMReader > &cdmReader, const **std::string** &outputFile, const **std::string** &configFile)
- virtual ~GribApiCDMWriter_Impl1 ()
- virtual void setParameter (const **std::string** &varName, const FimexTime &fTime, double levelValue) throw (CDMException)
- virtual void setProjection (const **std::string** &varName) throw (CDMException)
- virtual void setLevel (const **std::string** &varName, double levelValue)
- virtual boost::shared_ptr< Data > handleTypeScaleAndMissingData (const **std::string** &varName, const FimexTime &fTime, double levelValue, boost::shared_ptr< Data > inData)

15.45.1 Detailed Description

Implemenationn of a writer using GribApi for grib1

15.45.2 Constructor & Destructor Documentation

15.45.2.1 MetNoFimex::GribApiCDMWriter_Impl1::GribApiCDMWriter_Impl1 (const boost::shared_ptr< CDMReader > & *cdmReader*, const std::string & *outputFile*, const std::string & *configFile*)

15.45.2.2 virtual MetNoFimex::GribApiCDMWriter_Impl1::~GribApiCDMWriter_Impl1 () [virtual]

15.45.3 Member Function Documentation

15.45.3.1 virtual boost::shared_ptr<Data> MetNoFimex::GribApiCDMWriter_Impl1::handleTypeScaleAndMissingData (const std::string & *varName*, const FimexTime & *fTime*, double *levelValue*, boost::shared_ptr< Data > *inData*) [virtual]

add the missing value to the gribHandle, rescale the data if needed and change the datatype if needed, change the missingValue of the data if need

Returns

modified data

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

15.45.3.2 virtual void MetNoFimex::GribApiCDMWriter_Impl1::setLevel (const std::string & *varName*, double *levelValue*) [virtual]

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

15.45.3.3 virtual void MetNoFimex::GribApiCDMWriter_Impl1::setParameter (const std::string & *varName*, const FimexTime & *fTime*, double *levelValue*) throw (CDMException) [virtual]

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

15.45.3.4 virtual void MetNoFimex::GribApiCDMWriter_Impl1::setProjection (const std::string & *varName*) throw (CDMException) [virtual]

set the projection parameters, throw an exception if none are available

Parameters

varName

Exceptions

CDMException if parameters cannot be set

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

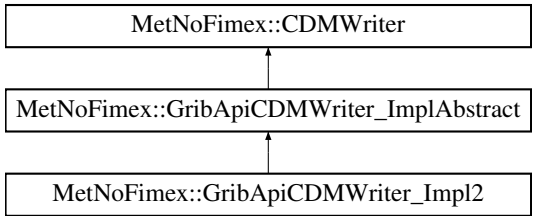
The documentation for this class was generated from the following file:

- include/fimex/GribApiCDMWriter_Impl1.h

15.46 MetNoFimex::GribApiCDMWriter_Impl2 Class Reference

`#include <GribApiCDMWriter_Impl2.h>`

Inheritance diagram for MetNoFimex::GribApiCDMWriter_Impl2:



Public Member Functions

- GribApiCDMWriter_Impl2 (const boost::shared_ptr< CDMReader > &cdmReader, const **std::string** &outputFile, const **std::string** &configFile)
- virtual ~GribApiCDMWriter_Impl2 ()
- virtual void setParameter (const **std::string** &varName, const FimexTime &fTime, double levelValue) throw (CDMException)
- virtual void setProjection (const **std::string** &varName) throw (CDMException)
- virtual void setLevel (const **std::string** &varName, double levelValue)
- virtual boost::shared_ptr< Data > handleTypeScaleAndMissingData (const **std::string** &varName, const FimexTime &fTime, double levelValue, boost::shared_ptr< Data > inData)

15.46.1 Detailed Description

Implemenationn of a writer using GribApi for grib2

15.46.2 Constructor & Destructor Documentation

15.46.2.1 MetNoFimex::GribApiCDMWriter_Impl2::GribApiCDMWriter_Impl2 (const boost::shared_ptr< CDMReader > & *cdmReader*, const std::string & *outputFile*, const std::string & *configFile*)

15.46.2.2 virtual MetNoFimex::GribApiCDMWriter_Impl2::~GribApiCDMWriter_Impl2 () [virtual]

15.46.3 Member Function Documentation

15.46.3.1 virtual boost::shared_ptr<Data> MetNoFimex::GribApiCDMWriter_Impl2::handleTypeScaleAndMissingData (const std::string & *varName*, const FimexTime & *fTime*, double *levelValue*, boost::shared_ptr< Data > *inData*) [virtual]

add the missing value to the gribHandle, rescale the data if needed and change the datatype if needed, change the missingValue of the data if need

Returns

modified data

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

15.46.3.2 virtual void MetNoFimex::GribApiCDMWriter_Impl2::setLevel (const std::string & *varName*, double *levelValue*) [virtual]

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

15.46.3.3 virtual void MetNoFimex::GribApiCDMWriter_Impl2::setParameter (const std::string & *varName*, const FimexTime & *fTime*, double *levelValue*) throw (CDMException) [virtual]

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

15.46.3.4 virtual void MetNoFimex::GribApiCDMWriter_Impl2::setProjection (const std::string & *varName*) throw (CDMException) [virtual]

set the projection parameters, throw an exception if none are available

Parameters

varName

Exceptions

CDMException if parameters cannot be set

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

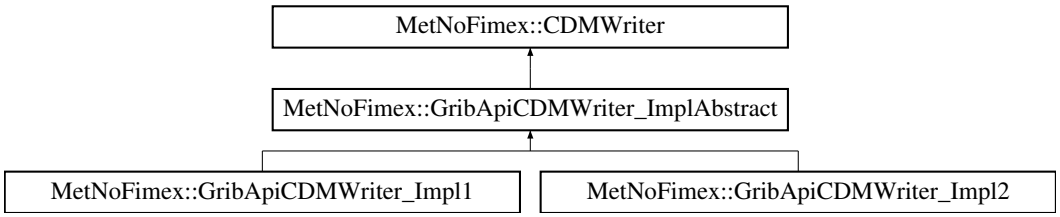
The documentation for this class was generated from the following file:

- include/fimex/GribApiCDMWriter_Impl2.h

15.47 MetNoFimex::GribApiCDMWriter_ImplAbstract Class Reference

`#include <GribApiCDMWriter_ImplAbstract.h>`

Inheritance diagram for MetNoFimex::GribApiCDMWriter_ImplAbstract:



Public Member Functions

- GribApiCDMWriter_ImplAbstract (int gribVersion, const boost::shared_ptr< CDMReader > &cdmReader, const **std::string** &outputFile, const **std::string** &configFile)
- virtual ~GribApiCDMWriter_ImplAbstract ()
- void run () throw (CDMException)

  actually write the data

Protected Member Functions

- virtual void setGlobalAttributes ()
- virtual void setData (const boost::shared_ptr< Data > &data)
- virtual void setProjection (const **std::string** &varName)=0 throw (CDMException)
- virtual void setParameter (const **std::string** &varName, const FimexTime &fTime, double levelValue)=0 throw (CDMException)
- virtual void setTime (const **std::string** &varName, const FimexTime &fTime)
- virtual void setLevel (const **std::string** &varName, double levelValue)=0
- virtual **std::vector**< double > getLevels (const **std::string** &varName) throw (CDMException)
- virtual **std::vector**< FimexTime > getTimes (const **std::string** &varName) throw (CDMException)
- virtual boost::shared_ptr< Data > handleTypeScaleAndMissingData (const **std::string** &varName, const FimexTime &fTime, double levelValue, boost::shared_ptr< Data > inData)=0
- virtual void writeGribHandleToFile ()
- xmlNode * getNodePtr (const **std::string** &varName, const FimexTime &fTime, double levelValue) throw (CDMException)

Protected Attributes

- int gribVersion
- const **std::string** configFile
- const boost::shared_ptr< XMLDoc > xmlConfig
- boost::shared_ptr< grib_handle > gribHandle
- LoggerPtr logger

15.47.1 Constructor & Destructor Documentation

15.47.1.1 MetNoFimex::GribApiCDMWriter_ImplAbstract::GribApiCDMWriter_ImplAbstract (int *gribVersion*, const boost::shared_ptr< CDMReader > & *cdmReader*, const std::string & *outputFile*, const std::string & *configFile*)

Constructor of the general writer. It should be called during construction of derived classes.

remember to call run to actually do something

15.47.1.2 virtual MetNoFimex::GribApiCDMWriter_ImplAbstract::~GribApiCDMWriter_ImplAbstract () [virtual]

15.47.2 Member Function Documentation

15.47.2.1 virtual std::vector<double> MetNoFimex::GribApiCDMWriter_ImplAbstract::getLevels (const std::string & *varName*) throw (CDMException) [protected, virtual]

get the levels from the cdm scaled to values used in grib (units/scale-factor) assign at least 1 level, give it a default value if none is found in the cdm

15.47.2.2 xmlNode* MetNoFimex::GribApiCDMWriter_ImplAbstract::getNodePtr (const std::string & *varName*, const FimexTime & *fTime*, double *levelValue*) throw (CDMException) [protected]

get the node belonging to varName, level and time from the config file

Parameters

varName name of the variable

fTime current time

level curent level

15.47.2.3 virtual std::vector<FimexTime> MetNoFimex::GribApiCDMWriter_ImplAbstract::getTimes (const std::string & *varName*) throw (CDMException) [protected, virtual]

get the times from the cdm as FimexTime (including unit) assign at least 1 time, give it a default value if none is found in the cdm

15.47.2.4 virtual boost::shared_ptr<Data> MetNoFimex::GribApiCDMWriter_ImplAbstract::handleTypeScaleAndMissingData (const std::string & *varName*, const FimexTime & *fTime*, double *levelValue*, boost::shared_ptr< Data > *inData*) [protected, pure virtual]

add the missing value to the gribHandle, rescale the data if needed and change the datatype if needed, change the missingValue of the data if need

Returns

modified data

Implemented in MetNoFimex::GribApiCDMWriter_Impl1, and MetNoFimex::GribApiCDMWriter_-Impl2.

15.47.2.5 void MetNoFimex::GribApiCDMWriter_ImplAbstract::run () throw (CDMException)

actually write the data

The run function has be to called after construction the object to actually fetch and write the data.

15.47.2.6 virtual void MetNoFimex::GribApiCDMWriter_ImplAbstract::setData (const boost::shared_ptr< Data > & *data*) [protected, virtual]

15.47.2.7 virtual void MetNoFimex::GribApiCDMWriter_ImplAbstract::setGlobalAttributes () [protected, virtual]

add the global attributes from the config to the default grib-handle

15.47.2.8 virtual void MetNoFimex::GribApiCDMWriter_ImplAbstract::setLevel (const std::string & *varName*, double *levelValue*) [protected, pure virtual]

Implemented in MetNoFimex::GribApiCDMWriter_Impl1, and MetNoFimex::GribApiCDMWriter_-Impl2.

15.47.2.9 virtual void MetNoFimex::GribApiCDMWriter_ImplAbstract::setParameter (const std::string & *varName*, const FimexTime & *fTime*, double *levelValue*) throw (CDMException) [protected, pure virtual]

Implemented in MetNoFimex::GribApiCDMWriter_Impl1, and MetNoFimex::GribApiCDMWriter_-Impl2.

15.47.2.10 virtual void MetNoFimex::GribApiCDMWriter_ImplAbstract::setProjection (const std::string & *varName*) throw (CDMException) [protected, pure virtual]

set the projection parameters, throw an exception if none are available

Parameters

varName

Exceptions

CDMException if parameters cannot be set

Implemented in MetNoFimex::GribApiCDMWriter_Impl1, and MetNoFimex::GribApiCDMWriter_-Impl2.

15.47.2.11 virtual void MetNoFimex::GribApiCDMWriter_ImplAbstract::setTime (const std::string & *varName*, const FimexTime & *fTime*) [protected, virtual]

15.47.2.12 virtual void MetNoFimex::GribApiCDMWriter_ImplAbstract::writeGribHandleToFile () [protected, virtual]

15.47.3 Member Data Documentation

15.47.3.1 const std::string MetNoFimex::GribApiCDMWriter_ImplAbstract::configFile [protected]

15.47.3.2 boost::shared_ptr<grib_handle> MetNoFimex::GribApiCDMWriter_ImplAbstract::gribHandle [protected]

15.47.3.3 int MetNoFimex::GribApiCDMWriter_ImplAbstract::gribVersion [protected]

15.47.3.4 LoggerPtr MetNoFimex::GribApiCDMWriter_ImplAbstract::logger [protected]

15.47.3.5 const boost::shared_ptr<XMLDoc> MetNoFimex::GribApiCDMWriter_ImplAbstract::xmlConfig [protected]

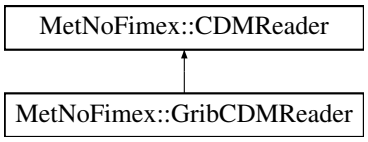
The documentation for this class was generated from the following file:

- include/fimex/GribApiCDMWriter_ImplAbstract.h

15.48 MetNoFimex::GribCDMReader Class Reference

`#include <GribCDMReader.h>`

Inheritance diagram for MetNoFimex::GribCDMReader:



Public Member Functions

- GribCDMReader (const **std::vector**< **std::string** > &fileNames, const XMLInput &configXML)
- virtual ~GribCDMReader ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos)

  data-reading function to be called from the CDMWriter

15.48.1 Constructor & Destructor Documentation

15.48.1.1 MetNoFimex::GribCDMReader::GribCDMReader (const std::vector< std::string > & *fileNames*, const XMLInput & *configXML*)

15.48.1.2 virtual MetNoFimex::GribCDMReader::~GribCDMReader () `[virtual]`

15.48.2 Member Function Documentation

15.48.2.1 virtual boost::shared_ptr<Data> MetNoFimex::GribCDMReader::getDataSlice (const std::string & *varName*, size_t *unLimDimPos*) `[virtual]`

data-reading function to be called from the CDMWriter

This methods needs to be implemented by the CDMReader. It should provide the data for each variable, either by reading from disk, converting from another CDMReader or reading from an in-memory data-section.

This function should retrieve the whole data for a dataset without unlimited dimension if the unLimDimPos == 0.

Parameters

varName name of the variable to read

unLimDimPos (optional) if the variable contains a unlimited dimension (max one allowed) an slice of this position is returned

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

Implements MetNoFimex::CDMReader.

The documentation for this class was generated from the following file:

- include/fimex/GribCDMReader.h

15.49 MetNoFimex::GribFileIndex Class Reference

`#include <GribFileIndex.h>`

Public Member Functions

- GribFileIndex ()
- GribFileIndex (boost::filesystem::path gribFilePath, bool ignoreExistingXml=false)
- virtual ~GribFileIndex ()
- const **std::vector**< GribFileMessage > & listMessages () const
- const **std::string** & getUrl () const

15.49.1 Constructor & Destructor Documentation

15.49.1.1 MetNoFimex::GribFileIndex::GribFileIndex ()

15.49.1.2 MetNoFimex::GribFileIndex::GribFileIndex (boost::filesystem::path *gribFilePath*, bool *ignoreExistingXml* = `false`)

Initialize the gribFileIndex for the gribFile gribFilePath. If ignoreExistingXml = false, searches for existing indexes in

- file.grbml
- ENV{GRIB_FILE_INDEX}/file.grbml

Otherwise, it parses the grib-file and creates a index in memory.

Performance for getting an index of a 150MB grib-file with some 10s of messages:

- remote NFS file, first time: 16s
- file completely in memory: 1.1s
- xml-file: 0.1s

15.49.1.3 virtual MetNoFimex::GribFileIndex::~GribFileIndex () `[virtual]`

15.49.2 Member Function Documentation

15.49.2.1 const std::string& MetNoFimex::GribFileIndex::getUrl () const `[inline]`

15.49.2.2 const std::vector<GribFileMessage>& MetNoFimex::GribFileIndex::listMessages () const `[inline]`

The documentation for this class was generated from the following file:

- include/fimex/GribFileIndex.h

15.50 MetNoFimex::GribFileMessage Class Reference

`#include <GribFileIndex.h>`

Public Member Functions

- GribFileMessage ()
- GribFileMessage (boost::shared_ptr< grib_handle > gh, const **std::string** &fileURL, long filePos, long msgPos)
- GribFileMessage (boost::shared_ptr< XMLDoc >, **std::string** nsPrefix, xmlNodePtr node)
- ~GribFileMessage ()
- bool isValid () const

  test if this is a proper GribFileMessage or just the default constructor

- **std::string** toString () const

  give a xml-string representation

- const long getEdition () const

  accessors

- const **std::string** & getFileURL () const
- const size_t getFilePosition () const
- const size_t getMessageNumber () const

  messages number within a multi-message

- const **std::string** & getName () const
- const **std::string** & getShortName () const
- boost::posix_time::ptime getValidTime () const
- boost::posix_time::ptime getReferenceTime () const
- long getLevelNumber () const
- long getLevelType () const
- const **std::vector**< long > & getParameterIds () const
- const **std::string** & getTypeOfGrid () const
- const GridDefinition & getGridDefinition () const
- size_t readData (**std::vector**< double > &data, double missingValue) const

15.50.1 Constructor & Destructor Documentation

15.50.1.1 MetNoFimex::GribFileMessage::GribFileMessage ()

15.50.1.2 MetNoFimex::GribFileMessage::GribFileMessage (boost::shared_ptr< grib_handle > *gh*, const std::string & *fileURL*, long *filePos*, long *msgPos*)

15.50.1.3 MetNoFimex::GribFileMessage::GribFileMessage (boost::shared_ptr< XMLDoc >, std::string *nsPrefix*, xmlNodePtr *node*)

15.50.1.4 MetNoFimex::GribFileMessage::~GribFileMessage ()

15.50.2 Member Function Documentation

15.50.2.1 const long MetNoFimex::GribFileMessage::getEdition () const

accessors

Referenced by MetNoFimex::GribFileMessageEqualLevelTime::operator()().

15.50.2.2 const size_t MetNoFimex::GribFileMessage::getFilePosition () const

15.50.2.3 const std::string& MetNoFimex::GribFileMessage::getFileURL () const

15.50.2.4 const GridDefinition& MetNoFimex::GribFileMessage::getGridDefinition () const

15.50.2.5 long MetNoFimex::GribFileMessage::getLevelNumber () const

Referenced by MetNoFimex::GribFileMessageEqualLevelTime::operator()().

15.50.2.6 long MetNoFimex::GribFileMessage::getLevelType () const

Referenced by MetNoFimex::GribFileMessageEqualLevelTime::operator()().

15.50.2.7 const size_t MetNoFimex::GribFileMessage::getMessageNumber () const

messages number within a multi-message

15.50.2.8 const std::string& MetNoFimex::GribFileMessage::getName () const

15.50.2.9 const std::vector<long>& MetNoFimex::GribFileMessage::getParameterIds () const

Get the parameter ids as list with the following meanings:

- ed1: indicatorOfParameter, gribTablesVersionNo, identificationOfOriginatingGeneratingCentre;
- ed2: parameterNumber, paramterCategory, discipline

15.50.2.10 boost::posix_time::ptime MetNoFimex::GribFileMessage::getReferenceTime () const

15.50.2.11 const std::string& MetNoFimex::GribFileMessage::getShortName () const

15.50.2.12 const std::string& MetNoFimex::GribFileMessage::getTypeOfGrid () const

15.50.2.13 boost::posix_time::ptime MetNoFimex::GribFileMessage::getValidTime () const

Referenced by MetNoFimex::GribFileMessageEqualLevelTime::operator()(), and MetNoFimex::GribFileMessageEqualTime::operator()().

15.50.2.14 bool MetNoFimex::GribFileMessage::isValid () const [inline]

test if this is a proper GribFileMessage or just the default constructor

15.50.2.15 size_t MetNoFimex::GribFileMessage::readData (std::vector< double > & *data*, double *missingValue*) const

Read the data from the underlying source to the vector data. Data of at maximum data.size() will be read.

Parameters

data the storage the data will be read to

missingValue the missing- / fill-value the returned data will have

Returns

the actual amount of data read

15.50.2.16 std::string MetNoFimex::GribFileMessage::toString () const

give a xml-string representation

The documentation for this class was generated from the following file:

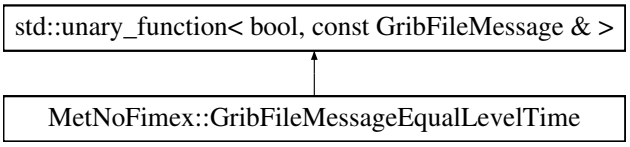
- include/fimex/GribFileIndex.h

15.51 MetNoFimex::GribFileMessageEqualLevelTime Class Reference

Functor to find messages with equal level and time.

`#include <GribFileIndex.h>`

Inheritance diagram for MetNoFimex::GribFileMessageEqualLevelTime:



Public Member Functions

- GribFileMessageEqualLevelTime (long edition, long levelType, long levelNo, boost::posix_time::ptime time)
- ~GribFileMessageEqualLevelTime ()
- bool operator() (const GribFileMessage &gfm)

15.51.1 Detailed Description

Functor to find messages with equal level and time.

15.51.2 Constructor & Destructor Documentation

15.51.2.1 MetNoFimex::GribFileMessageEqualLevelTime::GribFileMessageEqualLevelTime (long *edition*, long *levelType*, long *levelNo*, boost::posix_time::ptime *time*) [inline]

15.51.2.2 MetNoFimex::GribFileMessageEqualLevelTime::~GribFileMessageEqualLevelTime () [inline]

15.51.3 Member Function Documentation

15.51.3.1 bool MetNoFimex::GribFileMessageEqualLevelTime::operator() (const GribFileMessage & *gfm*) [inline]

References MetNoFimex::GribFileMessage::getEdition(), MetNoFimex::GribFileMessage::getLevelNumber(), MetNoFimex::GribFileMessage::getLevelType(), and MetNoFimex::GribFileMessage::getValidTime().

The documentation for this class was generated from the following file:

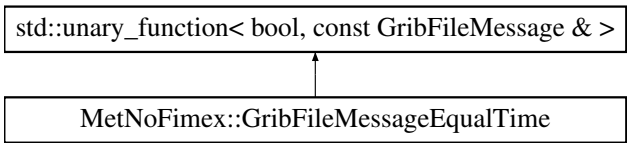
- include/fimex/GribFileIndex.h

15.52 MetNoFimex::GribFileMessageEqualTime Class Reference

Functor to find Messages with equal time.

`#include <GribFileIndex.h>`

Inheritance diagram for MetNoFimex::GribFileMessageEqualTime:



Public Member Functions

- GribFileMessageEqualTime (boost::posix_time::ptime time)
- ~GribFileMessageEqualTime ()
- bool operator() (const GribFileMessage &gfm)

15.52.1 Detailed Description

Functor to find Messages with equal time.

15.52.2 Constructor & Destructor Documentation

15.52.2.1 MetNoFimex::GribFileMessageEqualTime::GribFileMessageEqualTime (boost::posix_time::ptime *time*) [inline]

15.52.2.2 MetNoFimex::GribFileMessageEqualTime::~GribFileMessageEqualTime () [inline]

15.52.3 Member Function Documentation

15.52.3.1 bool MetNoFimex::GribFileMessageEqualTime::operator() (const GribFileMessage & *gfm*) [inline]

References MetNoFimex::GribFileMessage::getValidTime().

The documentation for this class was generated from the following file:

- include/fimex/GribFileIndex.h

15.53 MetNoFimex::GridDefinition Class Reference

`#include <GridDefinition.h>`

Public Types

- enum OrientationFlags { ScanStartRight = binary<010000000>::value, ScanStartBottom = binary<001000000>::value, ScanIsVertical = binary<000100000>::value, ScanIsAlternating = binary<000010000>::value }
- enum Orientation {

  LeftUpperHorizontal = binary<000000000>::value, RightUpperHorizontal = binary<010000000>::value, LeftLowerHorizontal = binary<001000000>::value, RightLowerHorizontal = binary<011000000>::value,

  LeftUpperVertical = binary<000100000>::value, RightUpperVertical = binary<010100000>::value, LeftLowerVertical = binary<001100000>::value, RightLowerVertical = binary<011100000>::value,

  LeftUpperHorizontalAlternating = binary<000010000>::value, RightUpperHorizontalAlternating = binary<010010000>::value, LeftLowerHorizontalAlternating = binary<001010000>::value, RightLowerHorizontalAlternating = binary<011010000>::value,

  LeftUpperVerticalAlternating = binary<000110000>::value, RightUpperVerticalAlternating = binary<010110000>::value, LeftLowerVerticalAlternating = binary<001110000>::value, RightLowerVerticalAlternating = binary<011110000>::value }

Public Member Functions

- GridDefinition ()
- GridDefinition (**std::string** projDefinition, size_t xSize, size_t ySize, double xIncr, double yIncr, double xStart, double yStart, Orientation orient)
- virtual ~GridDefinition ()
- virtual **std::string** getProjDefinition () const

  return a proj4 string

- virtual void setProjDefinition (**std::string** proj)
- virtual size_t getXSize () const

  number of points in x or longitude direction

- virtual void setXSize (size_t xSize)
- virtual size_t getYSize () const

  number of points in y or latitude direction

- virtual void setYSize (size_t ySize)
- virtual double getXIncrement () const

  x or longitude increment in m or degree

- virtual void setXIncrement (double xIncr)
- virtual double getYIncrement () const

  y or latitude increment in m or degree

- virtual void setYIncrement (double yIncr)

- virtual double getXStart () const

  x or longitude start in m or degree

- virtual void setXStart (double startX)
- virtual double getYStart () const

  y or latitude start in m or degree

- virtual void setYStart (double startY)
- virtual Orientation getScanMode () const
- virtual void setScanMode (Orientation orient)
- virtual bool comparableTo (const GridDefinition &rhs, double delta=0.) const

15.53.1 Member Enumeration Documentation

15.53.1.1 enum MetNoFimex::GridDefinition::Orientation

Enumerator:

LeftUpperHorizontal

RightUpperHorizontal

LeftLowerHorizontal

RightLowerHorizontal

LeftUpperVertical

RightUpperVertical

LeftLowerVertical

RightLowerVertical

LeftUpperHorizontalAlternating

RightUpperHorizontalAlternating

LeftLowerHorizontalAlternating

RightLowerHorizontalAlternating

LeftUpperVerticalAlternating

RightUpperVerticalAlternating

LeftLowerVerticalAlternating

RightLowerVerticalAlternating

15.53.1.2 enum MetNoFimex::GridDefinition::OrientationFlags

Use these flags to build an Orientation. Left, Upper, Horizontal and not Alternating are defaults and don't require flags.

Enumerator:

ScanStartRight

ScanStartBottom

ScanIsVertical

ScanIsAlternating change direction between succeeding rows (horizontal) or columns (vertical)

15.53.2 Constructor & Destructor Documentation

15.53.2.1 MetNoFimex::GridDefinition::GridDefinition ()

15.53.2.2 MetNoFimex::GridDefinition::GridDefinition (std::string *projDefinition*, size_t *xSize*, size_t *ySize*, double *xIncr*, double *yIncr*, double *xStart*, double *yStart*, Orientation *orient*)

15.53.2.3 virtual MetNoFimex::GridDefinition::~GridDefinition () `[virtual]`

15.53.3 Member Function Documentation

15.53.3.1 virtual bool MetNoFimex::GridDefinition::comparableTo (const GridDefinition & *rhs*, double *delta* = `0.`) const `[virtual]`

Compare two GridDefinitions. They are comparable if they have

- same size(XY)
- same incr(XY) within the delta
- same start(XY) within the delta

Parameters

rhs the other gridDefinition

delta the relative delta to compare to (a == 0) ? (abs(b) <= delta) : abs((b-a)/a) <= delta

15.53.3.2 virtual std::string MetNoFimex::GridDefinition::getProjDefinition () const `[virtual]`

return a proj4 string

15.53.3.3 virtual Orientation MetNoFimex::GridDefinition::getScanMode () const `[virtual]`

15.53.3.4 virtual double MetNoFimex::GridDefinition::getXIncrement () const `[virtual]`

x or longitude increment in m or degree

15.53.3.5 virtual size_t MetNoFimex::GridDefinition::getXSize () const `[virtual]`

number of points in x or longitude direction

15.53.3.6 virtual double MetNoFimex::GridDefinition::getXStart () const `[virtual]`

x or longitude start in m or degree

15.53.3.7 virtual double MetNoFimex::GridDefinition::getYIncrement () const `[virtual]`

y or latitude increment in m or degree

15.53.3.8 virtual size_t MetNoFimex::GridDefinition::getYSize () const [virtual]

number of points in y or latitude direction

15.53.3.9 virtual double MetNoFimex::GridDefinition::getYStart () const [virtual]

y or latitude start in m or degree

15.53.3.10 virtual void MetNoFimex::GridDefinition::setProjDefinition (std::string *proj*) [virtual]

15.53.3.11 virtual void MetNoFimex::GridDefinition::setScanMode (Orientation *orient*) [virtual]

15.53.3.12 virtual void MetNoFimex::GridDefinition::setXIncrement (double *xIncr*) [virtual]

15.53.3.13 virtual void MetNoFimex::GridDefinition::setXSize (size_t *xSize*) [virtual]

15.53.3.14 virtual void MetNoFimex::GridDefinition::setXStart (double *startX*) [virtual]

15.53.3.15 virtual void MetNoFimex::GridDefinition::setYIncrement (double *yIncr*) [virtual]

15.53.3.16 virtual void MetNoFimex::GridDefinition::setYSize (size_t *ySize*) [virtual]

15.53.3.17 virtual void MetNoFimex::GridDefinition::setYStart (double *startY*) [virtual]

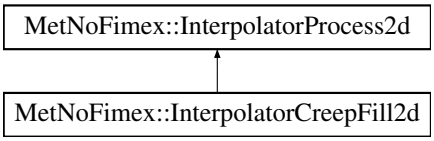
The documentation for this class was generated from the following file:

- include/fimex/GridDefinition.h

15.54 MetNoFimex::InterpolatorCreepFill2d Class Reference

`#include <CDMInterpolator.h>`

Inheritance diagram for MetNoFimex::InterpolatorCreepFill2d:



Public Member Functions

- InterpolatorCreepFill2d (unsigned short repeat, char setWeight)
- virtual void operator() (float *array, size_t nx, size_t ny)

15.54.1 Constructor & Destructor Documentation

15.54.1.1 MetNoFimex::InterpolatorCreepFill2d::InterpolatorCreepFill2d (unsigned short *repeat*, char *setWeight*) [inline]

15.54.2 Member Function Documentation

15.54.2.1 virtual void MetNoFimex::InterpolatorCreepFill2d::operator() (float * *array*, size_t *nx*, size_t *ny*) [inline, virtual]

Implements MetNoFimex::InterpolatorProcess2d.

References mifi_creepfill2d_f().

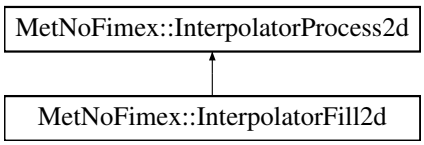
The documentation for this class was generated from the following file:

- include/fimex/CDMInterpolator.h

15.55 MetNoFimex::InterpolatorFill2d Class Reference

`#include <CDMInterpolator.h>`

Inheritance diagram for MetNoFimex::InterpolatorFill2d:



Public Member Functions

- InterpolatorFill2d (float relaxCrit, float corrEff, size_t maxLoop)
- virtual void operator() (float ∗array, size_t nx, size_t ny)

15.55.1 Constructor & Destructor Documentation

15.55.1.1 MetNoFimex::InterpolatorFill2d::InterpolatorFill2d (float *relaxCrit*, float *corrEff*, size_t *maxLoop*) [inline]

15.55.2 Member Function Documentation

15.55.2.1 virtual void MetNoFimex::InterpolatorFill2d::operator() (float ∗ *array*, size_t *nx*, size_t *ny*) [inline, virtual]

Implements MetNoFimex::InterpolatorProcess2d.

References mifi_fill2d_f().

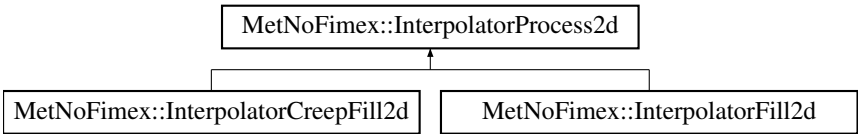
The documentation for this class was generated from the following file:

- include/fimex/CDMInterpolator.h

15.56 MetNoFimex::InterpolatorProcess2d Class Reference

`#include <CDMInterpolator.h>`

Inheritance diagram for MetNoFimex::InterpolatorProcess2d:



Public Member Functions

- virtual void operator() (float *array, size_t nx, size_t ny)=0
- virtual ~InterpolatorProcess2d ()

15.56.1 Detailed Description

operator interface to work on 2d arrays of size nx*ny

15.56.2 Constructor & Destructor Documentation

15.56.2.1 virtual MetNoFimex::InterpolatorProcess2d::~InterpolatorProcess2d () [inline, virtual]

15.56.3 Member Function Documentation

15.56.3.1 virtual void MetNoFimex::InterpolatorProcess2d::operator() (float * *array*, size_t *nx*, size_t *ny*) [pure virtual]

Implemented in MetNoFimex::InterpolatorFill2d, and MetNoFimex::InterpolatorCreepFill2d.

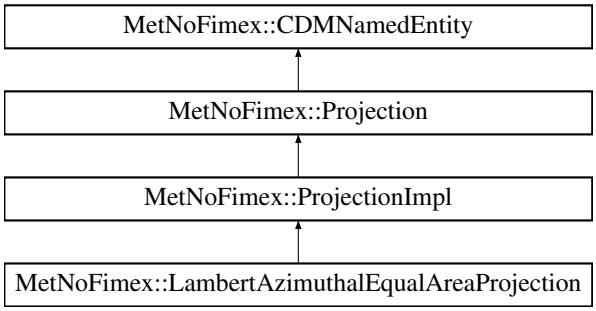
The documentation for this class was generated from the following file:

- include/fimex/CDMInterpolator.h

15.57 MetNoFimex::LambertAzimuthalEqualAreaProjection Class Reference

`#include <LambertAzimuthalEqualAreaProjection.h>`

Inheritance diagram for MetNoFimex::LambertAzimuthalEqualAreaProjection:



Public Member Functions

- LambertAzimuthalEqualAreaProjection ()
- virtual ~LambertAzimuthalEqualAreaProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- LambertAzimuthalEqualAreaProjection (**std::string** name)
- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.57.1 Constructor & Destructor Documentation

15.57.1.1 MetNoFimex::LambertAzimuthalEqualAreaProjection::LambertAzimuthalEqualAreaProjection ()

15.57.1.2 virtual MetNoFimex::LambertAzimuthalEqualAreaProjection::~LambertAzimuthalEqualAreaProjection () [inline, virtual]

15.57.1.3 MetNoFimex::LambertAzimuthalEqualAreaProjection::LambertAzimuthalEqualAreaProjection (std::string *name*) [inline, protected]

15.57.2 Member Function Documentation

15.57.2.1 static bool MetNoFimex::LambertAzimuthalEqualAreaProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

15.57.2.2 virtual std::ostream& MetNoFimex::LambertAzimuthalEqualAreaProjection::getProj4ProjectionPart (std::ostream &) const [protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.57.2.3 static std::vector<CDMAttribute> MetNoFimex::LambertAzimuthalEqualAreaProjection::parametersFromProj4 (const std::string & *proj4*) [static]

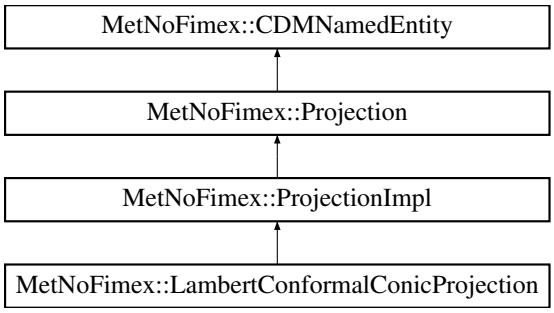
The documentation for this class was generated from the following file:

- include/fimex/coordSys/LambertAzimuthalEqualAreaProjection.h

15.58 MetNoFimex::LambertConformalConicProjection Class Reference

`#include <LambertConformalConicProjection.h>`

Inheritance diagram for MetNoFimex::LambertConformalConicProjection:



Public Member Functions

- LambertConformalConicProjection ()
- virtual ~LambertConformalConicProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.58.1 Constructor & Destructor Documentation

15.58.1.1 MetNoFimex::LambertConformalConicProjection::LambertConformalConicProjection ()

15.58.1.2 virtual MetNoFimex::LambertConformalConicProjection::~LambertConformalConicProjection () [inline, virtual]

15.58.2 Member Function Documentation

15.58.2.1 static bool MetNoFimex::LambertConformalConicProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

15.58.2.2 virtual std::ostream& MetNoFimex::LambertConformalConicProjection::getProj4ProjectionPart (std::ostream &) const [protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.58.2.3 static std::vector<CDMAttribute> MetNoFimex::LambertConformalConicProjection::parametersFromProj4 (const std::string & *proj4*) [static]

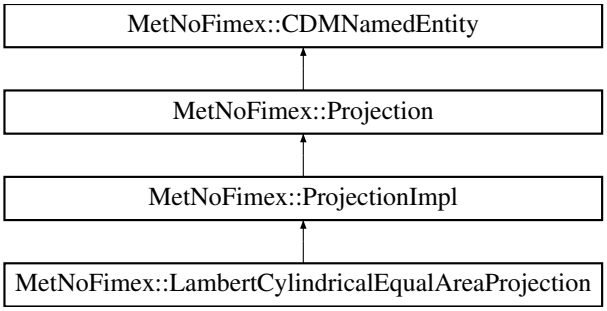
The documentation for this class was generated from the following file:

- include/fimex/coordSys/LambertConformalConicProjection.h

15.59 MetNoFimex::LambertCylindricalEqualAreaProjection Class Reference

`#include <LambertCylindricalEqualAreaProjection.h>`

Inheritance diagram for MetNoFimex::LambertCylindricalEqualAreaProjection:



Public Member Functions

- LambertCylindricalEqualAreaProjection ()
- virtual ~LambertCylindricalEqualAreaProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- LambertCylindricalEqualAreaProjection (**std::string** name)
- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.59.1 Constructor & Destructor Documentation

15.59.1.1 MetNoFimex::LambertCylindricalEqualAreaProjection::LambertCylindricalEqualAreaProjection ()

15.59.1.2 virtual MetNoFimex::LambertCylindricalEqualAreaProjection::~LambertCylindricalEqualAreaProjection () [inline, virtual]

15.59.1.3 MetNoFimex::LambertCylindricalEqualAreaProjection::LambertCylindricalEqualAreaProjection (std::string *name*) [inline, protected]

15.59.2 Member Function Documentation

15.59.2.1 static bool MetNoFimex::LambertCylindricalEqualAreaProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

15.59.2.2 virtual std::ostream& MetNoFimex::LambertCylindricalEqualAreaProjection::getProj4ProjectionPart (std::ostream &) const [protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.59.2.3 static std::vector<CDMAttribute> MetNoFimex::LambertCylindricalEqualAreaProjection::parametersFromProj4 (const std::string & *proj4*) [static]

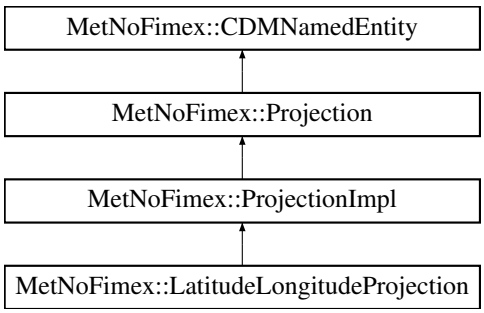
The documentation for this class was generated from the following file:

- include/fimex/coordSys/LambertCylindricalEqualAreaProjection.h

15.60 MetNoFimex::LatitudeLongitudeProjection Class Reference

`#include <LatitudeLongitudeProjection.h>`

Inheritance diagram for MetNoFimex::LatitudeLongitudeProjection:



Public Member Functions

- LatitudeLongitudeProjection ()
- virtual ~LatitudeLongitudeProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.60.1 Constructor & Destructor Documentation

15.60.1.1 MetNoFimex::LatitudeLongitudeProjection::LatitudeLongitudeProjection ()

15.60.1.2 virtual MetNoFimex::LatitudeLongitudeProjection::~LatitudeLongitudeProjection () [inline, virtual]

15.60.2 Member Function Documentation

15.60.2.1 static bool MetNoFimex::LatitudeLongitudeProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

15.60.2.2 virtual std::ostream& MetNoFimex::LatitudeLongitudeProjection::getProj4ProjectionPart (std::ostream &) const [inline, protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.60.2.3 static std::vector<CDMAttribute> MetNoFimex::LatitudeLongitudeProjection::parametersFromProj4 (const std::string & *proj4*) [static]

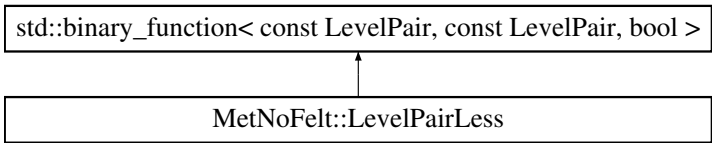
The documentation for this class was generated from the following file:

- include/fimex/coordSys/LatitudeLongitudeProjection.h

15.61 MetNoFelt::LevelPairLess Struct Reference

`#include <Felt_Types.h>`

Inheritance diagram for MetNoFelt::LevelPairLess:



Public Member Functions

- bool operator() (const **LevelPair** &p1, const **LevelPair** &p2) const

15.61.1 Detailed Description

comparison operator for pair<short, short> used for LevelPairs

15.61.2 Member Function Documentation

15.61.2.1 bool MetNoFelt::LevelPairLess::operator() (const LevelPair & *p1*, const LevelPair & *p2*) const [inline]

References std::pair< _T1, _T2 >::first, and std::pair< _T1, _T2 >::second.

The documentation for this struct was generated from the following file:

- include/fimex/Felt_Types.h

15.62 MetNoFimex::Logger Class Reference

`#include <Logger.h>`

Public Types

- enum LogLevel {

  OFF = 1000, FATAL = 900, ERROR = 800, WARN = 700,

  INFO = 600, DEBUG = 500 }

Public Member Functions

- Logger (const **std::string** &className)
- virtual ~Logger ()
- virtual bool isEnabledFor (LogLevel level)
- virtual void forcedLog (LogLevel level, const **std::string** &message, const char *filename, unsigned int lineNumber)

15.62.1 Detailed Description

Interface and default (dummy) implementation for a logger. Don't use this class directly, but retrieve a pointer to it via the getLogger function and log with the LOG4FIMEX macro.

15.62.2 Member Enumeration Documentation

15.62.2.1 enum MetNoFimex::Logger::LogLevel

different log levels

Enumerator:

OFF

FATAL

ERROR

WARN

INFO

DEBUG

15.62.3 Constructor & Destructor Documentation

15.62.3.1 MetNoFimex::Logger::Logger (const std::string & *className*)

15.62.3.2 virtual MetNoFimex::Logger::~Logger () [virtual]

15.62.4 Member Function Documentation

15.62.4.1 virtual void MetNoFimex::Logger::forcedLog (LogLevel *level*, const std::string & *message*, const char * *filename*, unsigned int *lineNumber*) [virtual]

log (without checking) for this loglevel

Parameters

level log-level to log

message log-message

filename best retrieved with __FILE__

lineNumber best retrieved with __LINE__

15.62.4.2 virtual bool MetNoFimex::Logger::isEnabledFor (LogLevel *level*) [virtual]

check if the loglevel of this logger is active

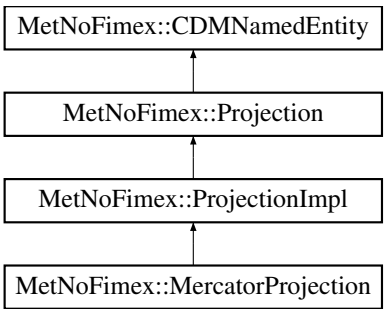
The documentation for this class was generated from the following file:

- include/fimex/Logger.h

15.63 MetNoFimex::MercatorProjection Class Reference

`#include <MercatorProjection.h>`

Inheritance diagram for MetNoFimex::MercatorProjection:



Public Member Functions

- MercatorProjection ()
- virtual ~MercatorProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.63.1 Constructor & Destructor Documentation

15.63.1.1 MetNoFimex::MercatorProjection::MercatorProjection ()

15.63.1.2 virtual MetNoFimex::MercatorProjection::~MercatorProjection () `[virtual]`

15.63.2 Member Function Documentation

15.63.2.1 static bool MetNoFimex::MercatorProjection::acceptsProj4 (const std::string & *proj4Str*) `[static]`

15.63.2.2 virtual std::ostream& MetNoFimex::MercatorProjection::getProj4ProjectionPart (std::ostream &) const `[protected, virtual]`

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.63.2.3 static std::vector<CDMAttribute> MetNoFimex::MercatorProjection::parametersFromProj4 (const std::string & *proj4*) [static]

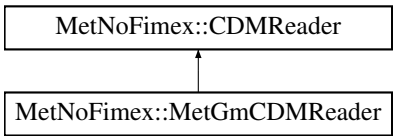
The documentation for this class was generated from the following file:

- include/fimex/coordSys/MercatorProjection.h

15.64 MetNoFimex::MetGmCDMReader Class Reference

`#include <MetGmCDMReader.h>`

Inheritance diagram for MetNoFimex::MetGmCDMReader:



Public Member Functions

- MetGmCDMReader (const **std::string** &metgmsource, const XMLInput &configXML)
- ~MetGmCDMReader ()
- boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos)

  data-reading function to be called from the CDMWriter

- boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, const SliceBuilder &sb)

  data-reading function to be called from the CDMWriter

15.64.1 Constructor & Destructor Documentation

15.64.1.1 MetNoFimex::MetGmCDMReader::MetGmCDMReader (const std::string & *metgmsource*, const XMLInput & *configXML*)

15.64.1.2 MetNoFimex::MetGmCDMReader::~MetGmCDMReader ()

15.64.2 Member Function Documentation

15.64.2.1 boost::shared_ptr<Data> MetNoFimex::MetGmCDMReader::getDataSlice (const std::string & *varName*, const SliceBuilder & *sb*) `[virtual]`

data-reading function to be called from the CDMWriter

Parameters

varName name of the variable to read

sb a SliceBuilder generated from this CDMReaders CDM

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions. This method has a default implementation depending on getDataSlice(varName, unLimDimPos), but should be implemented for performance reasons.

Reimplemented from MetNoFimex::CDMReader.

15.64.2.2 boost::shared_ptr<Data> MetNoFimex::MetGmCDMReader::getDataSlice (const std::string & *varName*, size_t *unLimDimPos*) [virtual]

data-reading function to be called from the CDMWriter

This methods needs to be implemented by the CDMReader. It should provide the data for each variable, either by reading from disk, converting from another CDMReader or reading from an in-memory data-section.

This function should retrieve the whole data for a dataset without unlimited dimension if the unLimDimPos == 0.

Parameters

varName name of the variable to read

unLimDimPos (optional) if the variable contains a unlimited dimension (max one allowed) an slice of this position is returned

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

Implements MetNoFimex::CDMReader.

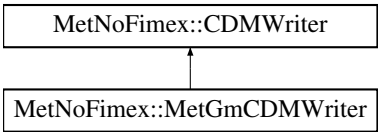
The documentation for this class was generated from the following file:

- include/fimex/MetGmCDMReader.h

15.65 MetNoFimex::MetGmCDMWriter Class Reference

`#include <MetGmCDMWriter.h>`

Inheritance diagram for MetNoFimex::MetGmCDMWriter:



Public Member Functions

- MetGmCDMWriter (boost::shared_ptr< CDMReader > cdmReader, const **std::string** &outputFile, const **std::string** &configFile=**std::string**())
- ~MetGmCDMWriter ()

Protected Attributes

- boost::shared_ptr< MetGmCDMWriterImpl > d_ptr

15.65.1 Constructor & Destructor Documentation

15.65.1.1 MetNoFimex::MetGmCDMWriter::MetGmCDMWriter (boost::shared_ptr< CDMReader > *cdmReader*, const std::string & *outputFile*, const std::string & *configFile* = std::string ( ))

15.65.1.2 MetNoFimex::MetGmCDMWriter::~MetGmCDMWriter ()

15.65.2 Member Data Documentation

15.65.2.1 boost::shared_ptr<MetGmCDMWriterImpl> MetNoFimex::MetGmCDMWriter::d_ptr [protected]

The documentation for this class was generated from the following file:

- include/fimex/MetGmCDMWriter.h

15.66 mifi_cdm_reader Class Reference

`#include <mifi_cdm_reader.h>`

Public Member Functions

- mifi_cdm_reader (boost::shared_ptr< MetNoFimex::CDMReader > reader)
- boost::shared_ptr< MetNoFimex::CDMReader > get ()

15.66.1 Detailed Description

wrapper class for boost::shared_ptr<CDMReader>, mainly for usage by C/C++ wrapper

15.66.2 Constructor & Destructor Documentation

15.66.2.1 mifi_cdm_reader::mifi_cdm_reader (boost::shared_ptr< MetNoFimex::CDMReader > *reader*) [inline]

15.66.3 Member Function Documentation

15.66.3.1 boost::shared_ptr<MetNoFimex::CDMReader> mifi_cdm_reader::get () [inline]

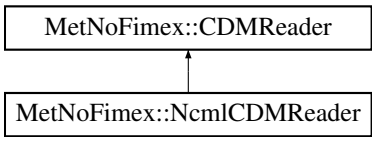
The documentation for this class was generated from the following file:

- include/fimex/mifi_cdm_reader.h

15.67 MetNoFimex::NcmlCDMReader Class Reference

`#include <NcmlCDMReader.h>`

Inheritance diagram for MetNoFimex::NcmlCDMReader:



Public Member Functions

- NcmlCDMReader (const XMLInput &configXML)
- NcmlCDMReader (const boost::shared_ptr< CDMReader > dataReader, const XMLInput &configXML)
- virtual ~NcmlCDMReader ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos=0)

15.67.1 Detailed Description

The NcmlCDMReader can be used as both standard reader of a data and as a manipulator for an existing CDM provided by a CDMReader.

In the case of a real reader, the ncml-configuration file needs to have the 'location' field set, which must point to a netcdf-file readable by NetCDF_CF10_CDMReader

The configuration file must be a standard ncml-file (versionn 2.2) as defined by `http://www.unidata.ucar.edu/software/netcdf/ncml/.`

Warning

The current version does not support aggregation.

15.67.2 Constructor & Destructor Documentation

15.67.2.1 MetNoFimex::NcmlCDMReader::NcmlCDMReader (const XMLInput & *configXML*)

Parameters

configXML ncml-file with location set

Exceptions

CDMException

15.67.2.2 MetNoFimex::NcmlCDMReader::NcmlCDMReader (const boost::shared_ptr< CDMReader > *dataReader*, const XMLInput & *configXML*)

Parameters

cdmReader a file reader opened elsewhere

configFile ncml-file with location set

Exceptions

CDMException

15.67.2.3 virtual MetNoFimex::NcmlCDMReader::~NcmlCDMReader () [virtual]

15.67.3 Member Function Documentation

15.67.3.1 virtual boost::shared_ptr<Data> MetNoFimex::NcmlCDMReader::getDataSlice (const std::string & *varName*, size_t *unLimDimPos* = 0) [virtual]

reading the data from the required source

Implements MetNoFimex::CDMReader.

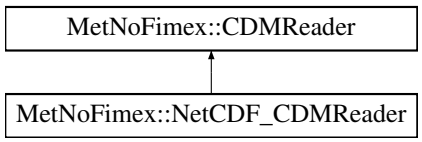
The documentation for this class was generated from the following file:

- include/fimex/NcmlCDMReader.h

15.68 MetNoFimex::NetCDF_CDMReader Class Reference

`#include "fimex/NetCDF_CDMReader.h"`

Inheritance diagram for MetNoFimex::NetCDF_CDMReader:



Public Member Functions

- NetCDF_CDMReader (const **std::string** &fileName)
- virtual ~NetCDF_CDMReader ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos)

  data-reading function to be called from the CDMWriter

- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, const SliceBuilder &sb)

  data-reading function to be called from the CDMWriter

15.68.1 Detailed Description

Examples:

coordinateSystem.cpp.

15.68.2 Constructor & Destructor Documentation

15.68.2.1 MetNoFimex::NetCDF_CDMReader::NetCDF_CDMReader (const std::string & *fileName*)

15.68.2.2 virtual MetNoFimex::NetCDF_CDMReader::~NetCDF_CDMReader () `[virtual]`

15.68.3 Member Function Documentation

15.68.3.1 virtual boost::shared_ptr<Data> MetNoFimex::NetCDF_CDMReader::getDataSlice (const std::string & *varName*, const SliceBuilder & *sb*) `[virtual]`

data-reading function to be called from the CDMWriter

Parameters

varName name of the variable to read

sb a SliceBuilder generated from this CDMReaders CDM

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions. This method has a default implementation depending on getDataSlice(varName, unLimDimPos), but should be implemented for performance reasons.

Reimplemented from MetNoFimex::CDMReader.

15.68.3.2 virtual boost::shared_ptr<Data> MetNoFimex::NetCDF_CDMReader::getDataSlice (const std::string & *varName*, size_t *unLimDimPos*) [virtual]

data-reading function to be called from the CDMWriter

This methods needs to be implemented by the CDMReader. It should provide the data for each variable, either by reading from disk, converting from another CDMReader or reading from an in-memory data-section.

This function should retrieve the whole data for a dataset without unlimited dimension if the unLimDimPos == 0.

Parameters

varName name of the variable to read

unLimDimPos (optional) if the variable contains a unlimited dimension (max one allowed) an slice of this position is returned

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

Implements MetNoFimex::CDMReader.

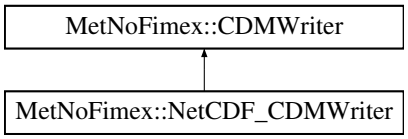
The documentation for this class was generated from the following file:

- include/fimex/NetCDF_CDMReader.h

15.69 MetNoFimex::NetCDF_CDMWriter Class Reference

`#include <NetCDF_CDMWriter.h>`

Inheritance diagram for MetNoFimex::NetCDF_CDMWriter:



Public Member Functions

- NetCDF_CDMWriter (const boost::shared_ptr< CDMReader > cdmReader, const **std::string** &outputFile, **std::string** configFile="", int version=3)
- virtual ~NetCDF_CDMWriter ()
- const **std::string** & getVariableName (const **std::string** &varName) const
- const **std::string** & getDimensionName (const **std::string** &dimName) const
- const CDMAttribute & getAttribute (const **std::string** &varName, const **std::string** &attName) const

15.69.1 Constructor & Destructor Documentation

15.69.1.1 MetNoFimex::NetCDF_CDMWriter::NetCDF_CDMWriter (const boost::shared_ptr< CDMReader > *cdmReader*, const std::string & *outputFile*, std::string *configFile* = "", int *version* = 3)

Parameters

cdmReader dataSource

outputFile file-name to write to

configFile xml-configuration

netcdf version, can be 3 or 4; 4 requires compilation against netcdf-4.0 or higher

15.69.1.2 virtual MetNoFimex::NetCDF_CDMWriter::~NetCDF_CDMWriter () `[virtual]`

15.69.2 Member Function Documentation

15.69.2.1 const CDMAttribute& MetNoFimex::NetCDF_CDMWriter::getAttribute (const std::string & *varName*, const std::string & *attName*) const

Warning

only public for testing

Parameters

varName original variable name (before config: newname)

attName original attribute name (before config: newname)

Returns

an attribute contained in the writers attribute, possibly added by config

15.69.2.2 const std::string& MetNoFimex::NetCDF_CDMWriter::getDimensionName (const std::string & *dimName*) const

Warning

only public for testing

Returns

the new name of a dimension, eventually changed by the writers config

15.69.2.3 const std::string& MetNoFimex::NetCDF_CDMWriter::getVariableName (const std::string & *varName*) const

Warning

only public for testing

Returns

the new name of a variable, eventually changed by the writers config

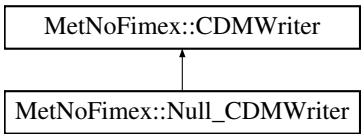
The documentation for this class was generated from the following file:

- include/fimex/NetCDF_CDMWriter.h

15.70 MetNoFimex::Null_CDMWriter Class Reference

`#include <Null_CDMWriter.h>`

Inheritance diagram for MetNoFimex::Null_CDMWriter:



Public Member Functions

- Null_CDMWriter (const boost::shared_ptr< CDMReader > cdmReader, const **std::string** &outputFile)
- virtual ~Null_CDMWriter ()

15.70.1 Detailed Description

CDMWriter does all operations as the NetCDF_CDMWriter, except writing to the file. This class is useful for performance tests.

15.70.2 Constructor & Destructor Documentation

15.70.2.1 MetNoFimex::Null_CDMWriter::Null_CDMWriter (const boost::shared_ptr< CDMReader > *cdmReader*, const std::string & *outputFile*)

15.70.2.2 virtual MetNoFimex::Null_CDMWriter::~Null_CDMWriter () `[virtual]`

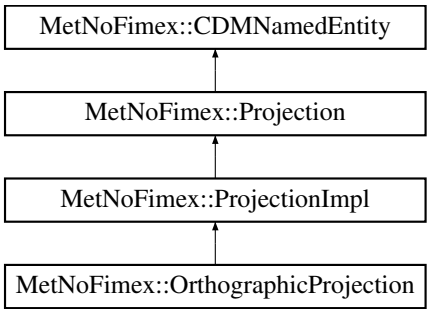
The documentation for this class was generated from the following file:

- include/fimex/Null_CDMWriter.h

15.71 MetNoFimex::OrthographicProjection Class Reference

`#include <OrthographicProjection.h>`

Inheritance diagram for MetNoFimex::OrthographicProjection:



Public Member Functions

- OrthographicProjection ()
- virtual ~OrthographicProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- OrthographicProjection (**std::string** name)
- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.71.1 Constructor & Destructor Documentation

15.71.1.1 MetNoFimex::OrthographicProjection::OrthographicProjection ()

15.71.1.2 virtual MetNoFimex::OrthographicProjection::~OrthographicProjection () [inline, virtual]

15.71.1.3 MetNoFimex::OrthographicProjection::OrthographicProjection (std::string *name*) [inline, protected]

15.71.2 Member Function Documentation

15.71.2.1 static bool MetNoFimex::OrthographicProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

15.71.2.2 virtual std::ostream& MetNoFimex::OrthographicProjection::getProj4ProjectionPart (std::ostream &) const [protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.71.2.3 static std::vector<CDMAttribute> MetNoFimex::OrthographicProjection::parametersFromProj4 (const std::string & *proj4*) [static]

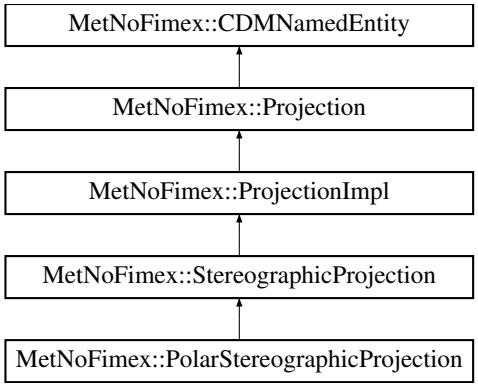
The documentation for this class was generated from the following file:

- include/fimex/coordSys/OrthographicProjection.h

15.72 MetNoFimex::PolarStereographicProjection Class Reference

`#include <PolarStereographicProjection.h>`

Inheritance diagram for MetNoFimex::PolarStereographicProjection:



Public Member Functions

- PolarStereographicProjection ()
- virtual ~PolarStereographicProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

15.72.1 Constructor & Destructor Documentation

15.72.1.1 MetNoFimex::PolarStereographicProjection::PolarStereographicProjection () [inline]

15.72.1.2 virtual MetNoFimex::PolarStereographicProjection::~PolarStereographicProjection () [inline, virtual]

15.72.2 Member Function Documentation

15.72.2.1 static bool MetNoFimex::PolarStereographicProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

Reimplemented from MetNoFimex::StereographicProjection.

15.72.2.2 static std::vector<CDMAttribute> MetNoFimex::PolarStereographicProjection::parametersFromProj4 (const std::string & *proj4*) [static]

Reimplemented from MetNoFimex::StereographicProjection.

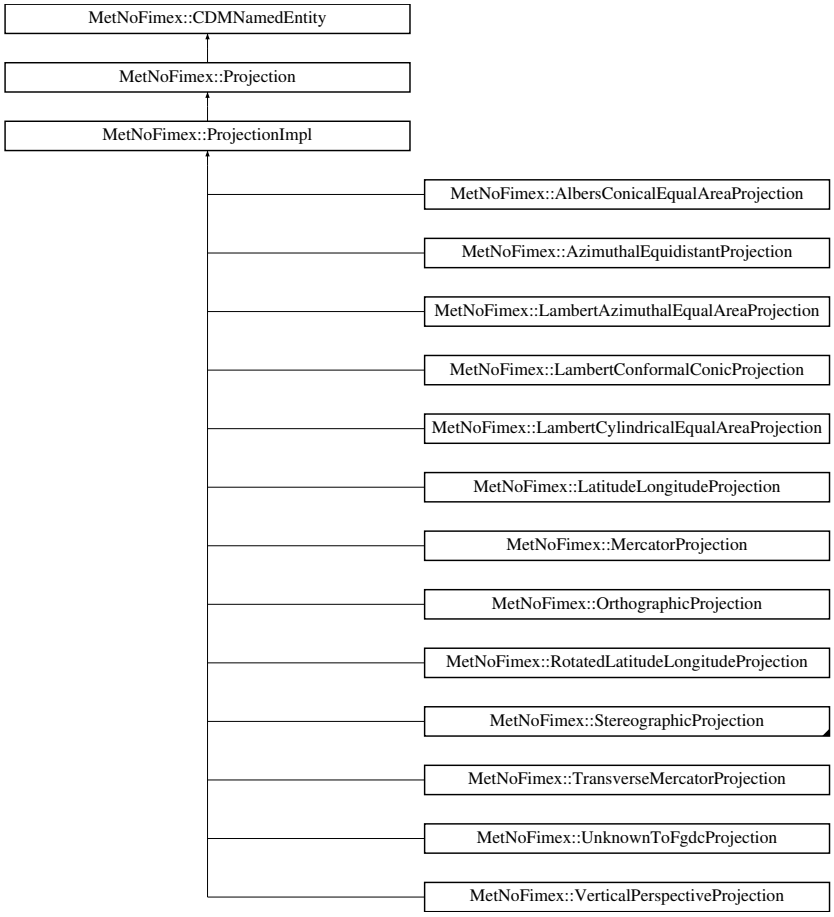
The documentation for this class was generated from the following file:

- include/fimex/coordSys/PolarStereographicProjection.h

15.73 MetNoFimex::Projection Class Reference

`#include <Projection.h>`

Inheritance diagram for MetNoFimex::Projection:



Public Member Functions

- virtual ~Projection ()
- virtual **std::vector**< CDMAttribute > getParameters () const =0
- virtual void addParameter (CDMAttribute attribute)=0
- virtual void addParameters (**std::vector**< CDMAttribute > attributes)=0
- virtual void removeParameter (**std::string** paramName)=0
- virtual const **std::string** & getName () const =0
- virtual const bool isDegree () const =0
- virtual **std::string** getProj4String () const =0
- virtual **std::string** getProj4EarthString () const =0
- virtual void convertToLonLat (**std::vector**< double > &xVals, **std::vector**< double > &yVals) const throw (CDMException)
- virtual void convertFromLonLat (**std::vector**< double > &xVals, **std::vector**< double > &yVals) const throw (CDMException)
- virtual **std::string** toString () const =0
- virtual bool operator== (const Projection &b) const

Static Public Member Functions

- static boost::shared_ptr< Projection > create (**std::vector**< CDMAttribute >)
- static boost::shared_ptr< Projection > createByProj4 (const **std::string** &projStr)

Protected Member Functions

- Projection ()

15.73.1 Detailed Description

A projection describes a projection of the earth surface from one system to another. If not mentioned otherwise, all parameters should follow CF-1.x

15.73.2 Constructor & Destructor Documentation

15.73.2.1 virtual MetNoFimex::Projection::~Projection () [inline, virtual]

15.73.2.2 MetNoFimex::Projection::Projection () [inline, protected]

15.73.3 Member Function Documentation

15.73.3.1 virtual void MetNoFimex::Projection::addParameter (CDMAttribute *attribute*) [pure virtual]

Implemented in MetNoFimex::ProjectionImpl.

15.73.3.2 virtual void MetNoFimex::Projection::addParameters (std::vector< CDMAttribute > *attributes*) [pure virtual]

Implemented in MetNoFimex::ProjectionImpl.

15.73.3.3 virtual void MetNoFimex::Projection::convertFromLonLat (std::vector< double > & *xVals*, std::vector< double > & *yVals*) const throw (CDMException) [virtual]

convert two vectors of values in the projection to latitude and longitude in degree. The earth ellipsoid will be the same as the input.

Parameters

xVals longitude input in degree, output of the projection axis in meter or degree

yVals latitude input in degree, output of the projection axis in meter or degree

Warning

the values will be converted in place

15.73.3.4 virtual void MetNoFimex::Projection::convertToLonLat (std::vector< double > & *xVals*, std::vector< double > & *yVals*) const throw (CDMException) [virtual]

convert two vectors of values in the projection to latitude and longitude in degree. The earth ellipsoid will be the same as the input.

Parameters

xVals input of the projection axis in meter or degree, output is longitude

yVals input of the projection axis in meter or degree, output is latitude

Warning

the values will be converted in place

15.73.3.5 static boost::shared_ptr<Projection> MetNoFimex::Projection::create (std::vector< CDMAttribute >) [static]

create a projection from some CDMAttributes

15.73.3.6 static boost::shared_ptr<Projection> MetNoFimex::Projection::createByProj4 (const std::string & *projStr*) [static]

create a projection from a proj4 string

15.73.3.7 virtual const std::string& MetNoFimex::Projection::getName () const [pure virtual]

get the projection name

Implements MetNoFimex::CDMNamedEntity.

Implemented in MetNoFimex::ProjectionImpl.

15.73.3.8 virtual std::vector<CDMAttribute> MetNoFimex::Projection::getParameters () const [pure virtual]

Implemented in MetNoFimex::ProjectionImpl.

15.73.3.9 virtual std::string MetNoFimex::Projection::getProj4EarthString () const [pure virtual]

get the parts of the proj4 string defining the earth.

Implemented in MetNoFimex::ProjectionImpl.

15.73.3.10 virtual std::string MetNoFimex::Projection::getProj4String () const [pure virtual]

get a proj4 string

Implemented in MetNoFimex::ProjectionImpl.

15.73.3.11 virtual const bool MetNoFimex::Projection::isDegree () const [pure virtual]

check if the coordinates belonging to this projection are in degree (otherwise metrical)

Implemented in MetNoFimex::ProjectionImpl.

15.73.3.12 virtual bool MetNoFimex::Projection::operator== (const Projection & *b*) const [virtual]

Comparison of two projections, implemented using the toString() function. This function does not guarantee that two projections are physically equal.

15.73.3.13 virtual void MetNoFimex::Projection::removeParameter (std::string *paramName*) [pure virtual]

Implemented in MetNoFimex::ProjectionImpl.

15.73.3.14 virtual std::string MetNoFimex::Projection::toString () const [pure virtual]

get a string representation

Note

this should be implemented as unique as possible, i.e.

Implemented in MetNoFimex::ProjectionImpl.

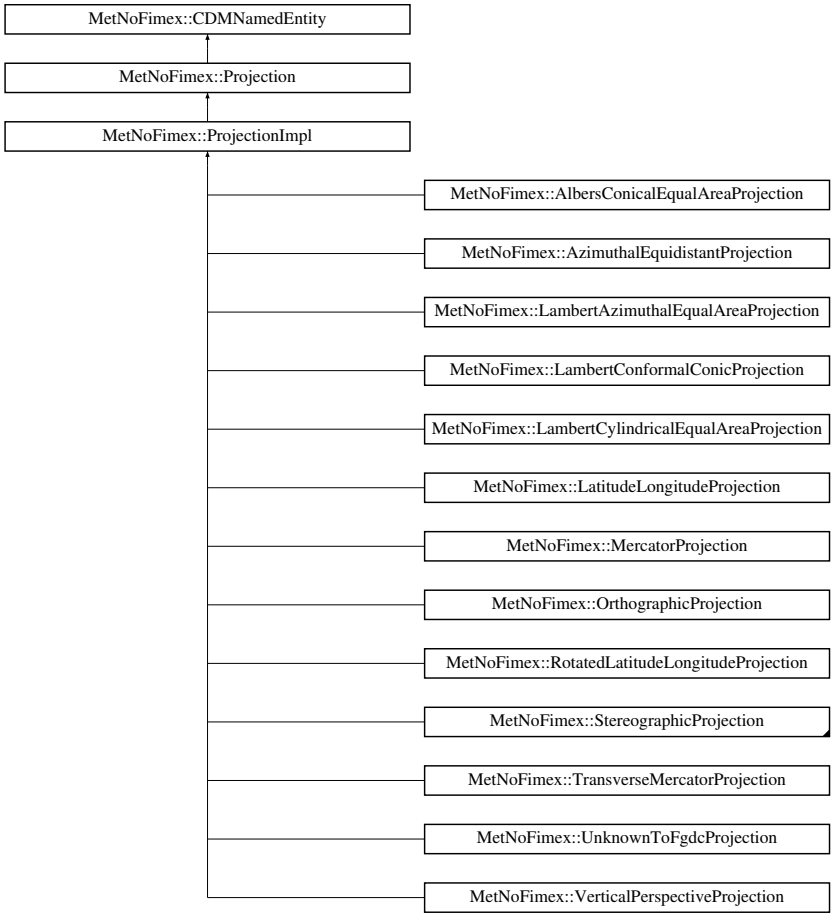
The documentation for this class was generated from the following file:

- include/fimex/coordSys/Projection.h

15.74 MetNoFimex::ProjectionImpl Class Reference

`#include <ProjectionImpl.h>`

Inheritance diagram for MetNoFimex::ProjectionImpl:



Public Member Functions

- virtual ~ProjectionImpl ()
- virtual **std::vector**< CDMAttribute > getParameters () const
- virtual void addParameter (CDMAttribute attribute)
- virtual void addParameters (**std::vector**< CDMAttribute > attributes)
- virtual void removeParameter (**std::string** paramName)
- virtual const **std::string** & getName () const
- virtual const bool isDegree () const
- virtual **std::string** getProj4String () const
- virtual **std::string** getProj4EarthString () const
- virtual **std::string** toString () const

Protected Member Functions

- ProjectionImpl (**std::string** name, bool isDegree)

- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &) const =0
- bool addParameterToStream (**std::ostream** &outStream, const **std::string** &name, **std::string** replaceName="") const

Static Protected Member Functions

- static bool proj4ProjectionMatchesName (const **std::string** &proj4String, const **std::string** &name)
- static void proj4GetEarthAttributes (const **std::string** &proj4String, **std::vector**< CDMAttribute > &attrList)

Protected Attributes

- **std::vector**< CDMAttribute > params_

15.74.1 Detailed Description

ProjectionImpl is a next to complete implementation of Projection, storing all parameters as a **vector**<**CDMAttribute**>. Implementations only need to implement a constructor calling the ProjectionImpl("proj-name") and the method getProj4ProjectionPart()

Note

the implemented projection needs to get made visible in Projection::create() and Projection::createByProj4

15.74.2 Constructor & Destructor Documentation

15.74.2.1 virtual MetNoFimex::ProjectionImpl::~ProjectionImpl () [virtual]

15.74.2.2 MetNoFimex::ProjectionImpl::ProjectionImpl (std::string *name*, bool *isDegree*) [explicit, protected]

15.74.3 Member Function Documentation

15.74.3.1 virtual void MetNoFimex::ProjectionImpl::addParameter (CDMAttribute *attribute*) [virtual]

Implements MetNoFimex::Projection.

15.74.3.2 virtual void MetNoFimex::ProjectionImpl::addParameters (std::vector< CDMAttribute > *attributes*) [virtual]

Implements MetNoFimex::Projection.

15.74.3.3 bool MetNoFimex::ProjectionImpl::addParameterToStream (std::ostream & *outStream*, const std::string & *name*, std::string *replaceName* = "") const [protected]

Add the numeric value of a parameter named name as replaceName to oproj, e.g. name = false_easting, replaceName = +x_0, sets "+x_0=... " Assume only one value at maximum

Parameters

outStream

name the parameters name

replaceName the name to use in the stream to the parameter, defaults to original name ("")

Returns

true if parameter found and set

15.74.3.4 virtual const std::string& MetNoFimex::ProjectionImpl::getName () const [virtual]

get the projection name

Implements MetNoFimex::Projection.

15.74.3.5 virtual std::vector<CDMAttribute> MetNoFimex::ProjectionImpl::getParameters () const [virtual]

Implements MetNoFimex::Projection.

15.74.3.6 virtual std::string MetNoFimex::ProjectionImpl::getProj4EarthString () const [virtual]

get the parts of the proj4 string defining the earth.

Implements MetNoFimex::Projection.

15.74.3.7 virtual std::ostream& MetNoFimex::ProjectionImpl::getProj4ProjectionPart (std::ostream &) const [protected, pure virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implemented in MetNoFimex::AlbersConicalEqualAreaProjection, MetNoFimex::AzimuthalEquidistantProjection, MetNoFimex::LambertAzimuthalEqualAreaProjection, MetNoFimex::LambertConformalConicProjection, MetNoFimex::LambertCylindricalEqualAreaProjection, MetNoFimex::LatitudeLongitudeProjection, MetNoFimex::MercatorProjection, MetNoFimex::OrthographicProjection, MetNoFimex::RotatedLatitudeLongitudeProjection, MetNoFimex::StereographicProjection, MetNoFimex::TransverseMercatorProjection, MetNoFimex::UnknownToFgdcProjection, and MetNoFimex::VerticalPerspectiveProjection.

15.74.3.8 virtual std::string MetNoFimex::ProjectionImpl::getProj4String () const [virtual]

get the proj4 string defined by the parameters. If a parameter named 'proj4' exists, that one will be used and all other parameters will be ignored.

Note

Implementors should not overwrite this method, but the protected getProj4ProjectionPart() method

Implements MetNoFimex::Projection.

15.74.3.9 virtual const bool MetNoFimex::ProjectionImpl::isDegree () const [virtual]

check if the coordinates belonging to this projection are in degree (otherwise metrical)

Implements MetNoFimex::Projection.

15.74.3.10 static void MetNoFimex::ProjectionImpl::proj4GetEarthAttributes (const std::string & *proj4String*, std::vector< CDMAttribute > & *attrList*) [static, protected]

add the attributes describing the earth from a proj4-string to the outAttrs

Parameters

proj4String string as used for proj4

output list of CDMAttributes

15.74.3.11 static bool MetNoFimex::ProjectionImpl::proj4ProjectionMatchesName (const std::string & *proj4String*, const std::string & *name*) [static, protected]

match the +proj= part of a proj4 string

15.74.3.12 virtual void MetNoFimex::ProjectionImpl::removeParameter (std::string *paramName*) [virtual]

Implements MetNoFimex::Projection.

15.74.3.13 virtual std::string MetNoFimex::ProjectionImpl::toString () const [virtual]

get a string representation

Implements MetNoFimex::Projection.

15.74.4 Member Data Documentation

15.74.4.1 std::vector<CDMAttribute> MetNoFimex::ProjectionImpl::params_ [protected]

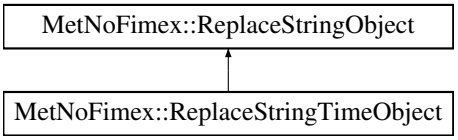
The documentation for this class was generated from the following file:

- include/fimex/coordSys/ProjectionImpl.h

15.75 MetNoFimex::ReplaceStringObject Class Reference

`#include <ReplaceStringObject.h>`

Inheritance diagram for MetNoFimex::ReplaceStringObject:



Public Member Functions

- virtual ~ReplaceStringObject ()=0
- virtual **std::ostream** & put (**std::ostream** &s) const =0
- virtual void setFormatString (const **std::string** &format)=0

  set the formatting string for this object

- virtual void setFormatStringAndOptions (const **std::string** &format, const **std::vector**< **std::string** > &options)=0

  set the formatting string and additional options for this object

15.75.1 Detailed Description

Interface for objects which might be converted to different strings

15.75.2 Constructor & Destructor Documentation

15.75.2.1 virtual MetNoFimex::ReplaceStringObject::~ReplaceStringObject () [pure virtual]

15.75.3 Member Function Documentation

15.75.3.1 virtual std::ostream& MetNoFimex::ReplaceStringObject::put (std::ostream & *s*) const [pure virtual]

put the formatted string to the stream

implementors are asked to implement operator<<

Implemented in MetNoFimex::ReplaceStringTimeObject.

15.75.3.2 virtual void MetNoFimex::ReplaceStringObject::setFormatString (const std::string & *format*) [pure virtual]

set the formatting string for this object

Implemented in MetNoFimex::ReplaceStringTimeObject.

15.75.3.3 virtual void MetNoFimex::ReplaceStringObject::setFormatStringAndOptions (const std::string & *format*, const std::vector< std::string > & *options*) [pure virtual]

set the formatting string and additional options for this object

Implemented in MetNoFimex::ReplaceStringTimeObject.

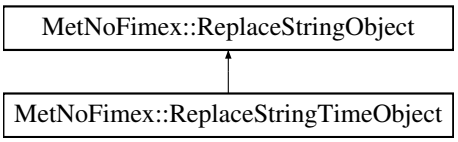
The documentation for this class was generated from the following file:

- include/fimex/ReplaceStringObject.h

15.76 MetNoFimex::ReplaceStringTimeObject Class Reference

`#include <ReplaceStringTimeObject.h>`

Inheritance diagram for MetNoFimex::ReplaceStringTimeObject:



Public Member Functions

- ReplaceStringTimeObject ()
- ReplaceStringTimeObject (std::time_t time, **std::string** format="%Y-%m-%d %H:%M:%S%F%Q")
- virtual ~ReplaceStringTimeObject ()
- virtual **std::ostream** & put (**std::ostream** &s) const
- virtual void setFormatString (const **std::string** &format)
- virtual void setFormatStringAndOptions (const **std::string** &format, const **std::vector**< **std::string** > &options)

Friends

- **std::ostream** & operator<< (**std::ostream** &s, const ReplaceStringTimeObject &rsto)

15.76.1 Constructor & Destructor Documentation

15.76.1.1 MetNoFimex::ReplaceStringTimeObject::ReplaceStringTimeObject () [inline]

15.76.1.2 MetNoFimex::ReplaceStringTimeObject::ReplaceStringTimeObject (std::time_t *time*, std::string *format* = "%Y-%m-%d %H:%M:%S%F%Q") [inline]

initialize a ReplaceStringTimeObject with time and string set

15.76.1.3 virtual MetNoFimex::ReplaceStringTimeObject::~ReplaceStringTimeObject () [inline, virtual]

15.76.2 Member Function Documentation

15.76.2.1 virtual std::ostream& MetNoFimex::ReplaceStringTimeObject::put (std::ostream & *s*) const [inline, virtual]

put the formatted string to the stream

implementors are asked to implement operator<<

Implements MetNoFimex::ReplaceStringObject.

15.76.2.2 virtual void MetNoFimex::ReplaceStringTimeObject::setFormatString (const std::string & *format*) [inline, virtual]

set the formatting String for this object

Parameters

format,: format string of strftime http://www.cplusplus.com/reference/clibrary/ctime/strftime.

Implements MetNoFimex::ReplaceStringObject.

15.76.2.3 virtual void MetNoFimex::ReplaceStringTimeObject::setFormatStringAndOptions (const std::string & *format*, const std::vector< std::string > & *options*) [virtual]

set the formatting string and additional options for this object options are: 0: offset as in seconds, i.e. +5000, -6000

Implements MetNoFimex::ReplaceStringObject.

15.76.3 Friends And Related Function Documentation

15.76.3.1 std::ostream& operator<< (std::ostream & *s*, const ReplaceStringTimeObject & *rsto*) [friend]

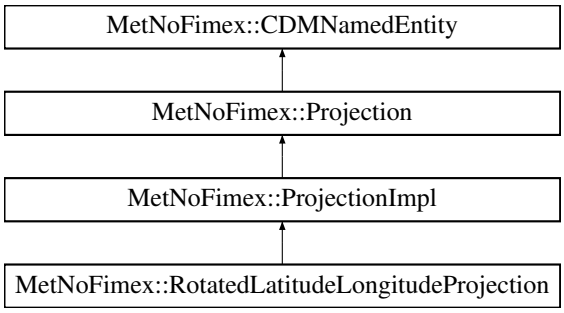
The documentation for this class was generated from the following file:

- include/fimex/ReplaceStringTimeObject.h

15.77 MetNoFimex::RotatedLatitudeLongitudeProjection Class Reference

`#include <RotatedLatitudeLongitudeProjection.h>`

Inheritance diagram for MetNoFimex::RotatedLatitudeLongitudeProjection:



Public Member Functions

- RotatedLatitudeLongitudeProjection ()
- virtual ~RotatedLatitudeLongitudeProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.77.1 Constructor & Destructor Documentation

15.77.1.1 MetNoFimex::RotatedLatitudeLongitudeProjection::RotatedLatitudeLongitudeProjection ()

15.77.1.2 virtual MetNoFimex::RotatedLatitudeLongitudeProjection::~RotatedLatitudeLongitudeProjection () [inline, virtual]

15.77.2 Member Function Documentation

15.77.2.1 static bool MetNoFimex::RotatedLatitudeLongitudeProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

15.77.2.2 virtual std::ostream& MetNoFimex::RotatedLatitudeLongitudeProjection::getProj4ProjectionPart (std::ostream &) const [protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.77.2.3 static std::vector<CDMAttribute> MetNoFimex::RotatedLatitudeLongitudeProjection::parametersFromProj4 (const std::string & *proj4*) [static]

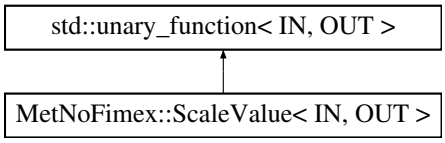
The documentation for this class was generated from the following file:

- include/fimex/coordSys/RotatedLatitudeLongitudeProjection.h

15.78 MetNoFimex::ScaleValue< IN, OUT > Class Template Reference

`#include <Utils.h>`

Inheritance diagram for MetNoFimex::ScaleValue< IN, OUT >:



Public Member Functions

- ScaleValue (double oldFill, double oldScale, double oldOffset, double newFill, double newScale, double newOffset)
- OUT operator() (const IN &in) const

15.78.1 Detailed Description

template<typename IN, typename OUT> class MetNoFimex::ScaleValue< IN, OUT >

Scale a value using fill, offset and scale

15.78.2 Constructor & Destructor Documentation

15.78.2.1 template<typename IN , typename OUT > MetNoFimex::ScaleValue< IN, OUT >::ScaleValue (double *oldFill*, double *oldScale*, double *oldOffset*, double *newFill*, double *newScale*, double *newOffset*) [inline]

15.78.3 Member Function Documentation

15.78.3.1 template<typename IN , typename OUT > OUT MetNoFimex::ScaleValue< IN, OUT >::operator() (const IN & *in*) const [inline]

The documentation for this class was generated from the following file:

- include/fimex/Utils.h

15.79 MetNoFimex::SharedArrayConstCastDeleter< T > Struct Template Reference

`#include <Utils.h>`

Public Member Functions

- SharedArrayConstCastDeleter (boost::shared_array< T > ptr)
- template<typename C >
  void operator() (C *)

Protected Attributes

- boost::shared_array< T > ptr

15.79.1 Detailed Description

template<typename T> struct MetNoFimex::SharedArrayConstCastDeleter< T >

delete-class for shared_array's, making sure that the original shared_array does not expire before the current shared_array. Use as

boost::shared_array<int> bla; boost::shared_array<const int>(bla.get(), SharedArrayConstCastDeleter(bla));

15.79.2 Constructor & Destructor Documentation

15.79.2.1 template<typename T > MetNoFimex::SharedArrayConstCastDeleter< T >::SharedArrayConstCastDeleter (boost::shared_array< T > *ptr*) [inline]

15.79.3 Member Function Documentation

15.79.3.1 template<typename T > template<typename C > void MetNoFimex::SharedArrayConstCastDeleter< T >::operator() (C *) [inline]

15.79.4 Member Data Documentation

15.79.4.1 template<typename T > boost::shared_array<T> MetNoFimex::SharedArrayConstCastDeleter< T >::ptr [protected]

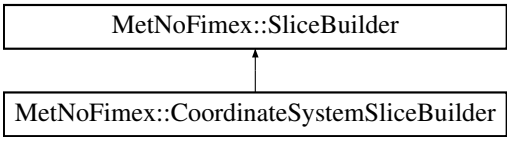
The documentation for this struct was generated from the following file:

- include/fimex/Utils.h

15.80 MetNoFimex::SliceBuilder Class Reference

`#include "fimex/SliceBuilder.h"`

Inheritance diagram for MetNoFimex::SliceBuilder:



Public Member Functions

- SliceBuilder (const CDM &cdm, const **std::string** &varName)
- SliceBuilder (const **std::vector**< **std::string** > &dimNames, const **std::vector**< size_t > &dimSize)
- virtual ~SliceBuilder ()
- void setStartAndSize (const **std::string** &dimName, size_t start, size_t size)
- void setStartAndSize (const boost::shared_ptr< const CoordinateAxis > &axis, size_t start, size_t size)
- void setAll (const **std::string** &dimName)
- void setAll (const boost::shared_ptr< const CoordinateAxis > &axis)
- const **std::vector**< size_t > & getDimensionStartPositions () const
- const **std::vector**< size_t > & getDimensionSizes () const
- **std::vector**< **std::string** > getDimensionNames () const
- **std::vector**< **std::string** > getUnsetDimensionNames () const
- const **std::vector**< size_t > & getMaxDimensionSizes () const

Protected Member Functions

- size_t getDimPos (const **std::string** &dimName) const

15.80.1 Constructor & Destructor Documentation

15.80.1.1 MetNoFimex::SliceBuilder::SliceBuilder (const CDM & *cdm*, const std::string & *varName*)

Create a new slice builder for variable varName. The default will give a slice of full size.

Parameters

cdm

varName variable name

Exceptions

CDMException f varName doesn't exists

15.80.1.2 MetNoFimex::SliceBuilder::SliceBuilder (const std::vector< std::string > & *dimNames*, const std::vector< size_t > & *dimSize*)

Simple interface to create a slicebuilder. No checks are made if the created object is useful with any reader.

Parameters

dimNames names of dimensions

dimSize maximum size of the dimensions

15.80.1.3 virtual MetNoFimex::SliceBuilder::~SliceBuilder () `[virtual]`

15.80.2 Member Function Documentation

15.80.2.1 std::vector<std::string> MetNoFimex::SliceBuilder::getDimensionNames () const

Returns

vector with names of dimensions in correct order

15.80.2.2 const std::vector<size_t>& MetNoFimex::SliceBuilder::getDimensionSizes () const `[inline]`

Returns

vector with sizes of the dimenions of the variable in the order and size of the variables dimensions

15.80.2.3 const std::vector<size_t>& MetNoFimex::SliceBuilder::getDimensionStartPositions () const `[inline]`

Returns

vector with start-positions of shape-size and order of the variable

15.80.2.4 size_t MetNoFimex::SliceBuilder::getDimPos (const std::string & *dimName*) const `[protected]`

15.80.2.5 const std::vector<size_t>& MetNoFimex::SliceBuilder::getMaxDimensionSizes () const `[inline]`

Returns

vector with maximum sizes of the dimenions of the variable in the order and size of the variables dimensions

15.80.2.6 std::vector<std::string> MetNoFimex::SliceBuilder::getUnsetDimensionNames () const

Returns

vector with names of dimensions which have not been set yet, (i.e. through setAll() or setStartAndSize()

Examples:

coordinateSystem.cpp.

15.80.2.7 void MetNoFimex::SliceBuilder::setAll (const boost::shared_ptr< const CoordinateAxis > & *axis*)

Set the start to 0 and the size to the maximum size. Though this is the default this function will reset prevous reduced dimensions and it will mark the dimension as 'set' and not return it in getUnsetDimensionNames.

Parameters

axis name of the dimension to restrict, ignored if NULL

15.80.2.8 void MetNoFimex::SliceBuilder::setAll (const std::string & *dimName*)

Set the start to 0 and the size to the maximum size. Though this is the default this function will reset prevous reduced dimensions and it will mark the dimension as 'set' and not return it in getUnsetDimensionNames.

Parameters

axis name of the dimension to restrict

Examples:

coordinateSystem.cpp.

15.80.2.9 void MetNoFimex::SliceBuilder::setStartAndSize (const boost::shared_ptr< const CoordinateAxis > & *axis*, size_t *start*, size_t *size*)

set the start position and the size of the slice of that dimension

Parameters

axis name of the dimension to restrict, ignored if NULL

start starting point of slice (starts at 0)

size size of the slice

Exceptions

CDMException if axis not part of the dimensions of the variable

out_of_range depending on startPos or size

15.80.2.10 void MetNoFimex::SliceBuilder::setStartAndSize (const std::string & *dimName*, size_t *start*, size_t *size*)

set the start position and the size of the slice of that dimension

Parameters

dimName name of the dimension to restrict

start starting point of slice (starts at 0)

size size of the slice

Exceptions

CDMException if dimName not part of the dimensions of variable

out_of_range depending on startPos or size

Examples:

coordinateSystem.cpp.

The documentation for this class was generated from the following file:

- include/fimex/SliceBuilder.h

15.81 MetNoFimex::SpatialAxisSpec Class Reference

`#include <SpatialAxisSpec.h>`

Public Member Functions

- SpatialAxisSpec (const **std::string** &axisSpec) throw (CDMException)
- SpatialAxisSpec (const **std::string** &axisSpec, double start, double end) throw (CDMException)
- virtual ~SpatialAxisSpec ()
- bool requireStartEnd ()
- void setStartEnd (double start, double end)
- const **std::vector**< double > & getAxisSteps ()

15.81.1 Detailed Description

This class can be used to describe a list of spatial units in an efficient textual way.

- UNIT: see udunit, compatible with degree or m: default: m
- RELVALUE: float-number
- RELVALUES: comma-separated list of values with possible ... extension, ... meaning continuation of the difference of the previous two values 0 is the first time in the original time-axis, x is the last time-value in the original time-axis

A SpatialAxisSpec consists of at least of values:

- axisspec := VALUES[;unit=UNIT] | RELVALUES;relativeStart=VALUE[;unit=UNIT]

relativeStart will reset the relative value 0 to the first value larger than x0 (original start time) with x0 = i ∗ (v1-v0)∗ unit with i being a integer.

```
axisspec = -450000,-400000,...,50000

timespec = -50,0,...,x,x+50;relativeStart=17;unit=km
```

Warning

The 'unit' parameter is currently not supported, please enter values as m or degree
the RELVALUES currently must be in m, degree not supported (yet?)

15.81.2 Constructor & Destructor Documentation

15.81.2.1 MetNoFimex::SpatialAxisSpec::SpatialAxisSpec (const std::string & *axisSpec*) throw (CDMException) `[inline]`

Define a spatialAxisSpec. Depending on the axisSpec (relativeStart?), start and end must be given later

Parameters

axisSpec string representation as explained above

15.81.2.2 MetNoFimex::SpatialAxisSpec::SpatialAxisSpec (const std::string & *axisSpec*, double *start*, double *end*) throw (CDMException) [inline]

Define a spatialAxisSpec

Parameters

axisSpec string representation as explained above

start place of data start, in degree or m

end place to end, in degree or m

15.81.2.3 virtual MetNoFimex::SpatialAxisSpec::~SpatialAxisSpec () [inline, virtual]

15.81.3 Member Function Documentation

15.81.3.1 const std::vector<double>& MetNoFimex::SpatialAxisSpec::getAxisSteps () [inline]

Returns

steps on the axis in degree or m

15.81.3.2 bool MetNoFimex::SpatialAxisSpec::requireStartEnd ()

Check if axisSpec still requires start and end place. This returns false if a) start and end have been given already b) the axisSpec is independant of start and end

15.81.3.3 void MetNoFimex::SpatialAxisSpec::setStartEnd (double *start*, double *end*) [inline]

The documentation for this class was generated from the following file:

- include/fimex/SpatialAxisSpec.h

15.82 MetNoFimex::staticCast< OUT > Struct Template Reference

`#include <Utils.h>`

Public Member Functions

- template<typename IN >
  OUT operator() (const IN &in)

15.82.1 Detailed Description

template<typename OUT> struct MetNoFimex::staticCast< OUT >

static_cast as a functor

15.82.2 Member Function Documentation

15.82.2.1 template<typename OUT > template<typename IN > OUT MetNoFimex::staticCast< OUT >::operator() (const IN & *in*) [inline]

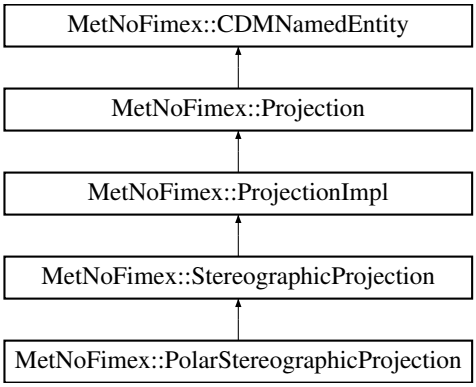
The documentation for this struct was generated from the following file:

- include/fimex/Utils.h

15.83 MetNoFimex::StereographicProjection Class Reference

`#include <StereographicProjection.h>`

Inheritance diagram for MetNoFimex::StereographicProjection:



Public Member Functions

- StereographicProjection ()
- virtual ~StereographicProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- StereographicProjection (**std::string** name)
- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.83.1 Constructor & Destructor Documentation

15.83.1.1 MetNoFimex::StereographicProjection::StereographicProjection ()

15.83.1.2 virtual MetNoFimex::StereographicProjection::~StereographicProjection () [inline, virtual]

15.83.1.3 MetNoFimex::StereographicProjection::StereographicProjection (std::string *name*) [inline, protected]

15.83.2 Member Function Documentation

15.83.2.1 static bool MetNoFimex::StereographicProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

Reimplemented in MetNoFimex::PolarStereographicProjection.

15.83.2.2 virtual std::ostream& MetNoFimex::StereographicProjection::getProj4ProjectionPart (std::ostream &) const [protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.83.2.3 static std::vector<CDMAttribute> MetNoFimex::StereographicProjection::parametersFromProj4 (const std::string & *proj4*) [static]

Reimplemented in MetNoFimex::PolarStereographicProjection.

The documentation for this class was generated from the following file:

- include/fimex/coordSys/StereographicProjection.h

15.84 MetNoFimex::TimeLevelDataSliceFetcher Class Reference

read a slice of a given time/level combination from a cdmReader

`#include <TimeLevelDataSliceFetcher.h>`

Public Member Functions

- TimeLevelDataSliceFetcher (boost::shared_ptr< CDMReader > cdmReader, const **std::string** &varName)
- virtual ~TimeLevelDataSliceFetcher ()
- boost::shared_ptr< Data > getTimeLevelSlice (size_t time, size_t level) throw (CDMException)

15.84.1 Detailed Description

read a slice of a given time/level combination from a cdmReader

15.84.2 Constructor & Destructor Documentation

15.84.2.1 MetNoFimex::TimeLevelDataSliceFetcher::TimeLevelDataSliceFetcher (boost::shared_ptr< CDMReader > *cdmReader*, const std::string & *varName*)

initialize the Fetcher

Parameters

cdmReader the reader to fetch the original data from

varName the variable to read the data from

15.84.2.2 virtual MetNoFimex::TimeLevelDataSliceFetcher::~TimeLevelDataSliceFetcher () `[virtual]`

15.84.3 Member Function Documentation

15.84.3.1 boost::shared_ptr<Data> MetNoFimex::TimeLevelDataSliceFetcher::getTimeLevelSlice (size_t *time*, size_t *level*) throw (CDMException)

get the slice of time at position time and level at position level join unlimited dimensions if needed, slice data if needed

Parameters

time the position of the time according to the variables level-dimension

level the position of the level according to the level-dimension

The documentation for this class was generated from the following file:

- include/fimex/TimeLevelDataSliceFetcher.h

15.85 MetNoFimex::TimeSpec Class Reference

`#include <TimeSpec.h>`

Public Member Functions

- TimeSpec (const **std::string** &timeSpec, const FimexTime &startTime, const FimexTime &endTime) throw (CDMException)
- virtual ~TimeSpec ()
- const **std::vector**< FimexTime > & getTimeSteps () const
- const **std::string** & getUnitString () const

15.85.1 Detailed Description

This class can be used to describe a list of times in an efficient textual way.

Unless otherwise mentioned, i.e. with *bounds* a value v(time) describes the time at exactly that instance. All times are UTC.

- TIMESTAMP format: YYYY-MM-DD HH:MM:SS
- TIMESTAMPS: comma-separated list of values with possible ... extension, ... meaning continuation of the difference of the previous two values
- UNIT: see `udunit`, default: second
- VALUE: float-number
- VALUES: comma-separated list of values with possible ... extension, ... meaning continuation of the difference of the previous two values 0 is the first time in the original time-axis, x is the last time-value in the original time-axis

A TimeSpec consists of at least of timestamps or values:

- timespec := (TIMESTAMPS | VALUES[;relativeUnit=UNIT])[;unit=UNIT]

relativeUnit will reset the relative value 0 to the first value larger than t0 (original start time) with t0 = i ∗ (v1-v0)∗ unit with i being a integer.

```
timespec = 2000-01-01 00:00:00,2000-01-01 00:04:00,...,2010-01-01 00:00:00
```

All times outside the original time-axis will be discarded.

```
timespec = -3,0,3,...,x,x+3;relativeUnit=hours since 2000-01-01 00:00:00;unit=ho
     urs since 2000-01-01 00:00:00;
```

15.85.2 Constructor & Destructor Documentation

15.85.2.1 MetNoFimex::TimeSpec::TimeSpec (const std::string & *timeSpec*, const FimexTime & *startTime*, const FimexTime & *endTime*) throw (CDMException)

Define a timeSpec

Parameters

timeSpec string representation as explained above

startTime time to start in case of a relativeStart timeSpec

endTime time to end in case of a relativeStart timeSpec

15.85.2.2 virtual MetNoFimex::TimeSpec::~TimeSpec () [inline, virtual]

15.85.3 Member Function Documentation

15.85.3.1 const std::vector<FimexTime>& MetNoFimex::TimeSpec::getTimeSteps () const [inline]

15.85.3.2 const std::string& MetNoFimex::TimeSpec::getUnitString () const [inline]

The documentation for this class was generated from the following file:

- include/fimex/TimeSpec.h

15.86 MetNoFimex::TimeUnit Class Reference

`#include <TimeUnit.h>`

Public Member Functions

- TimeUnit () throw (CDMException)

  initialize a timeUnit with a unit string

- TimeUnit (const **std::string** &timeUnitString) throw (CDMException)
- virtual ~TimeUnit ()
- double unitTime2epochSeconds (double unitTime) const

  calculate the epochSeconds for a time in the current unit

- boost::posix_time::ptime unitTime2posixTime (double unitTime) const

  calculate the ptime for a time in the current unit

- double epochSeconds2unitTime (double epochSeconds) const

  calculate the time in the current unit from the epoch

- FimexTime unitTime2fimexTime (double unitTime) const throw (CDMException)

  calculate the time in a calendar form

- double fimexTime2unitTime (const FimexTime &fiTime) const throw (CDMException)

  calculate the time in the current unit from the calendar form

- double fimexTime2unitTimeX (FimexTime fiTime) const throw (CDMException)

  *same as fimexTime2unitTime but copying fiTime instead of referencing, needed for i.e. bind1st(***mem_fun()***)*

- double posixTime2unitTime (boost::posix_time::ptime poTime) const throw (CDMException)

  calculate the unitTime from a boost::posix_time

15.86.1 Detailed Description

TimeUnit calculates times from a time given in a unit as of CF-1.0 (e.g. 'days since 2000-01-01 00:00:00') to a unix time (i.e. 'seconds since 1970-01-01 00:00:00') or a time struct MetNoFimex::FimexTime

All times are assumed to be UTC, and we use the Gregorian Calendar (not 100% true for times before 1600AD, depending on implementation)

15.86.2 Constructor & Destructor Documentation

15.86.2.1 MetNoFimex::TimeUnit::TimeUnit () throw (CDMException)

initialize a timeUnit with a unit string

15.86.2.2 MetNoFimex::TimeUnit::TimeUnit (const std::string & *timeUnitString*) throw (CDMException)

15.86.2.3 virtual MetNoFimex::TimeUnit::~TimeUnit () `[virtual]`

15.86.3 Member Function Documentation

15.86.3.1 double MetNoFimex::TimeUnit::epochSeconds2unitTime (double *epochSeconds*) const

calculate the time in the current unit from the epoch

15.86.3.2 double MetNoFimex::TimeUnit::fimexTime2unitTime (const FimexTime & *fiTime*) const throw (CDMException)

calculate the time in the current unit from the calendar form

Referenced by fimexTime2unitTimeX().

15.86.3.3 double MetNoFimex::TimeUnit::fimexTime2unitTimeX (FimexTime *fiTime*) const throw (CDMException) `[inline]`

same as fimexTime2unitTime but copying fiTime instead of referencing, needed for i.e. bind1st(**mem_fun**())

References fimexTime2unitTime().

15.86.3.4 double MetNoFimex::TimeUnit::posixTime2unitTime (boost::posix_time::ptime *poTime*) const throw (CDMException)

calculate the unitTime from a boost::posix_time

15.86.3.5 double MetNoFimex::TimeUnit::unitTime2epochSeconds (double *unitTime*) const

calculate the epochSeconds for a time in the current unit

15.86.3.6 FimexTime MetNoFimex::TimeUnit::unitTime2fimexTime (double *unitTime*) const throw (CDMException)

calculate the time in a calendar form

15.86.3.7 boost::posix_time::ptime MetNoFimex::TimeUnit::unitTime2posixTime (double *unitTime*) const

calculate the ptime for a time in the current unit

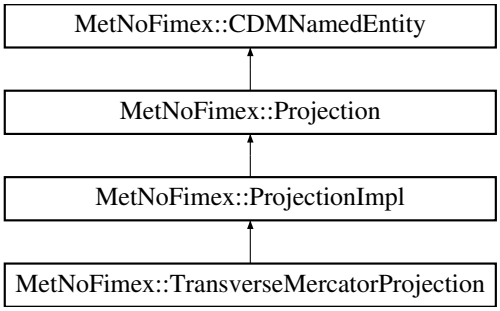
The documentation for this class was generated from the following file:

- include/fimex/TimeUnit.h

15.87 MetNoFimex::TransverseMercatorProjection Class Reference

`#include <TransverseMercatorProjection.h>`

Inheritance diagram for MetNoFimex::TransverseMercatorProjection:



Public Member Functions

- TransverseMercatorProjection ()
- virtual ~TransverseMercatorProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.87.1 Constructor & Destructor Documentation

15.87.1.1 MetNoFimex::TransverseMercatorProjection::TransverseMercatorProjection ()

15.87.1.2 virtual MetNoFimex::TransverseMercatorProjection::~TransverseMercatorProjection () [virtual]

15.87.2 Member Function Documentation

15.87.2.1 static bool MetNoFimex::TransverseMercatorProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

15.87.2.2 virtual std::ostream& MetNoFimex::TransverseMercatorProjection::getProj4ProjectionPart (std::ostream &) const [protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.87.2.3 static std::vector<CDMAttribute> MetNoFimex::TransverseMercatorProjection::parametersFromProj4 (const std::string & *proj4*) [static]

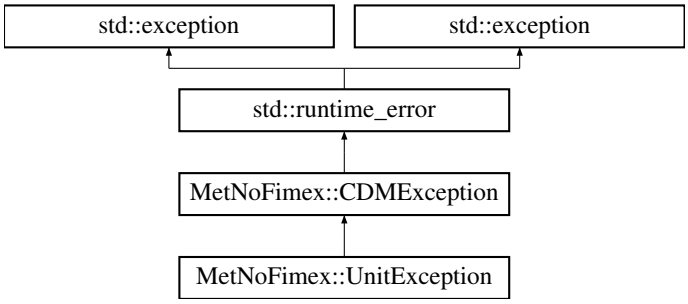
The documentation for this class was generated from the following file:

- include/fimex/coordSys/TransverseMercatorProjection.h

15.88 MetNoFimex::UnitException Class Reference

`#include <Units.h>`

Inheritance diagram for MetNoFimex::UnitException:



Public Member Functions

- UnitException ()
- UnitException (**std::string** message)

15.88.1 Constructor & Destructor Documentation

15.88.1.1 MetNoFimex::UnitException::UnitException () [inline]

15.88.1.2 MetNoFimex::UnitException::UnitException (std::string *message*) [inline]

The documentation for this class was generated from the following file:

- include/fimex/Units.h

15.89 MetNoFimex::Units Class Reference

`#include <Units.h>`

Public Member Functions

- Units ()
- Units (const Units &rhs)
- Units & operator= (const Units &rhs)
- virtual ~Units ()
- void convert (const **std::string** &from, const **std::string** &to, double &slope, double &offset) throw (UnitException)
- bool areConvertible (const **std::string** &unit1, const **std::string** &unit2) const

  test if two units are convertible to each others

- bool isTime (const **std::string** &timeUnit) const

  test if unit is a time

- const void * exposeInternals () const

Static Public Member Functions

- static bool unload (bool force=false) throw (UnitException)

15.89.1 Detailed Description

The class Units describes a units-system, not a single unit. Different units can be compared and converted if comparable within the system.

15.89.2 Constructor & Destructor Documentation

15.89.2.1 MetNoFimex::Units::Units ()

initialization of unit handling, i.e. parsing of unit file etc if required the unit file is installation-dependent on the underlying units-package (udunits or udunits2) and can be controlled through UDUNITS_PATH environment

15.89.2.2 MetNoFimex::Units::Units (const Units & *rhs*)

15.89.2.3 virtual MetNoFimex::Units::~Units () `[virtual]`

15.89.3 Member Function Documentation

15.89.3.1 bool MetNoFimex::Units::areConvertible (const std::string & *unit1*, const std::string & *unit2*) const

test if two units are convertible to each others

Parameters

unit1 first unit

unit2 second unit

15.89.3.2 void MetNoFimex::Units::convert (const std::string & *from*, const std::string & *to*, double & *slope*, double & *offset*) throw (UnitException)

calculate the linear unit conversion: newVal (in to unit) = oldVal (in from unit) ∗ slope + offset

Parameters

from unit

to unit

slope return value of the slope

offset return value of the offset

15.89.3.3 const void∗ MetNoFimex::Units::exposeInternals () const

expose the internals of the implementation as a void∗ you need to be sure that you know the internals!

Needed in TimeUnit.

15.89.3.4 bool MetNoFimex::Units::isTime (const std::string & *timeUnit*) const

test if unit is a time

Parameters

timeUnit

15.89.3.5 Units& MetNoFimex::Units::operator= (const Units & *rhs*)

15.89.3.6 static bool MetNoFimex::Units::unload (bool *force* = `false`) throw (UnitException) `[static]`

Units initialize themselve on first using the default unix-file path and keep the internal datastructure until the end of the program, or this function is used.

Parameters

force unload units-setup, even if some objects exist, defaults to false

Returns

true if unloaded, false if there are still some objects using the internal data-structure.

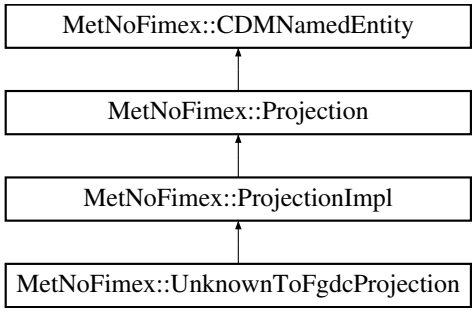
The documentation for this class was generated from the following file:

- include/fimex/Units.h

15.90 MetNoFimex::UnknownToFgdcProjection Class Reference

`#include <UnknownToFgdcProjection.h>`

Inheritance diagram for MetNoFimex::UnknownToFgdcProjection:



Public Member Functions

- UnknownToFgdcProjection ()
- virtual ~UnknownToFgdcProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- UnknownToFgdcProjection (**std::string** name)
- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.90.1 Constructor & Destructor Documentation

15.90.1.1 MetNoFimex::UnknownToFgdcProjection::UnknownToFgdcProjection ()

15.90.1.2 virtual MetNoFimex::UnknownToFgdcProjection::~UnknownToFgdcProjection () [inline, virtual]

15.90.1.3 MetNoFimex::UnknownToFgdcProjection::UnknownToFgdcProjection (std::string *name*) [inline, protected]

15.90.2 Member Function Documentation

15.90.2.1 static bool MetNoFimex::UnknownToFgdcProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

returns allways true

15.90.2.2 virtual std::ostream& MetNoFimex::UnknownToFgdcProjection::getProj4ProjectionPart (std::ostream &) const [protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.90.2.3 static std::vector<CDMAttribute> MetNoFimex::UnknownToFgdcProjection::parametersFromProj4 (const std::string & *proj4*) [static]

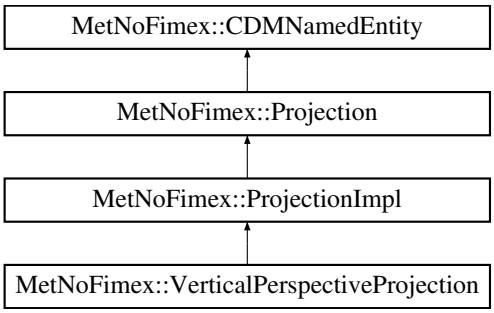
The documentation for this class was generated from the following file:

- include/fimex/coordSys/UnknownToFgdcProjection.h

15.91 MetNoFimex::VerticalPerspectiveProjection Class Reference

`#include <VerticalPerspectiveProjection.h>`

Inheritance diagram for MetNoFimex::VerticalPerspectiveProjection:



Public Member Functions

- VerticalPerspectiveProjection ()
- virtual ~VerticalPerspectiveProjection ()

Static Public Member Functions

- static bool acceptsProj4 (const **std::string** &proj4Str)
- static **std::vector**< CDMAttribute > parametersFromProj4 (const **std::string** &proj4)

Protected Member Functions

- VerticalPerspectiveProjection (**std::string** name)
- virtual **std::ostream** & getProj4ProjectionPart (**std::ostream** &oproj) const

15.91.1 Constructor & Destructor Documentation

15.91.1.1 MetNoFimex::VerticalPerspectiveProjection::VerticalPerspectiveProjection ()

15.91.1.2 virtual MetNoFimex::VerticalPerspectiveProjection::~VerticalPerspectiveProjection () [inline, virtual]

15.91.1.3 MetNoFimex::VerticalPerspectiveProjection::VerticalPerspectiveProjection (std::string *name*) [inline, protected]

15.91.2 Member Function Documentation

15.91.2.1 static bool MetNoFimex::VerticalPerspectiveProjection::acceptsProj4 (const std::string & *proj4Str*) [static]

15.91.2.2 virtual std::ostream& MetNoFimex::VerticalPerspectiveProjection::getProj4ProjectionPart (std::ostream &) const [protected, virtual]

add the pure projection parameters for proj4 to the stream, i.e. no earth definitions, and no +no_defs

Implements MetNoFimex::ProjectionImpl.

15.91.2.3 static std::vector<CDMAttribute> MetNoFimex::VerticalPerspectiveProjection::parametersFromProj4 (const std::string & *proj4*) [static]

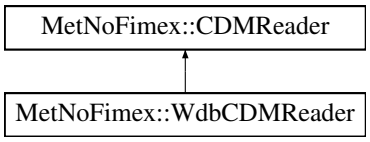
The documentation for this class was generated from the following file:

- include/fimex/coordSys/VerticalPerspectiveProjection.h

15.92 MetNoFimex::WdbCDMReader Class Reference

`#include <WdbCDMReader.h>`

Inheritance diagram for MetNoFimex::WdbCDMReader:



Public Member Functions

- WdbCDMReader (const **std::string** &source, const XMLInput &configXML)
- virtual ~WdbCDMReader ()
- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos)

  data-reading function to be called from the CDMWriter

- virtual boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, const SliceBuilder &sb)

  data-reading function to be called from the CDMWriter

15.92.1 Detailed Description

CDM reader for wdb databases.

Since wdb is a database system, a configuration file is used in place of a "real" data file. This configuration file tells how to connect to a wdb database, and what query to perform on it.

General syntax for the wdb query file may be found in the wdb_query.xsd file, with an annotated example in the wdb.example.wdbml file.

Also, it is possible to use a specification in place of the file. That specification is a colon- or semicolon separated list of name-value pairs.

It is possible to use a hybrid specification, in which you give additions to the queries in the query file. If you want to specify queries in this way you must follow the syntax, used in the following example:

"file=whatever.wdbml:dataprovider=whoever:referencetime=latest"

The following keywords are recognized: file, dbname, host, port, user, wciUser, dataprovider, location, referencetime, validtime, parameter and dataversion. They follow the same rule as the corresponding keywords in the xml-files.

Syntax for global configuration is given in wdb_conf.xsd.

Since there is much freedom in wdb, the generated CDMs from different wdb instances can be very different from each other. In general all dimensions of size one will be skipped in the resulting dimensions and variables.

15.92.2 Constructor & Destructor Documentation

15.92.2.1 MetNoFimex::WdbCDMReader::WdbCDMReader (const std::string & *source*, const XMLInput & *configXML*)

Specifications of where to find the database, and what query to run on it is given in the file with name source. Generic specifications are given in the file with name configfilename.

15.92.2.2 virtual MetNoFimex::WdbCDMReader::~WdbCDMReader () [virtual]

15.92.3 Member Function Documentation

15.92.3.1 virtual boost::shared_ptr<Data> MetNoFimex::WdbCDMReader::getDataSlice (const std::string & *varName*, const SliceBuilder & *sb*) [virtual]

data-reading function to be called from the CDMWriter

Parameters

varName name of the variable to read

sb a SliceBuilder generated from this CDMReaders CDM

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions. This method has a default implementation depending on getDataSlice(varName, unLimDimPos), but should be implemented for performance reasons.

Reimplemented from MetNoFimex::CDMReader.

15.92.3.2 virtual boost::shared_ptr<Data> MetNoFimex::WdbCDMReader::getDataSlice (const std::string & *varName*, size_t *unLimDimPos*) [virtual]

data-reading function to be called from the CDMWriter

This methods needs to be implemented by the CDMReader. It should provide the data for each variable, either by reading from disk, converting from another CDMReader or reading from an in-memory data-section.

This function should retrieve the whole data for a dataset without unlimited dimension if the unLimDimPos == 0.

Parameters

varName name of the variable to read

unLimDimPos (optional) if the variable contains a unlimited dimension (max one allowed) an slice of this position is returned

Exceptions

CDMException on errors related to the CDM in combination with the underlying data-structure. It might also throw other (IO-)exceptions.

Implements MetNoFimex::CDMReader.

The documentation for this class was generated from the following file:

- include/fimex/WdbCDMReader.h

15.93 MetNoFimex::XMLDoc Class Reference

`#include <XMLDoc.h>`

Public Member Functions

- XMLDoc (const **std::string** &filename)
- virtual ~XMLDoc ()
- XPathObjPtr getXPathObject (const **std::string** &xpath, xmlNodePtr node=0) const
- void registerNamespace (const **std::string** &prefix, const **std::string** &uri)

  register a namespace for later xpath

Static Public Member Functions

- static boost::shared_ptr< XMLDoc > fromFile (const **std::string** &filename)
- static boost::shared_ptr< XMLDoc > fromString (const **std::string** &buffer, const **std::string** &url="")
- static boost::shared_ptr< XMLDoc > fromURL (const **std::string** &url)

15.93.1 Detailed Description

a tiny wrapper around libxml dom and xpath reader with xml::include

15.93.2 Constructor & Destructor Documentation

15.93.2.1 MetNoFimex::XMLDoc::XMLDoc (const std::string & *filename*) `[explicit]`

initialization of libxml and the xml config file

Parameters

filename xml input-file

Exceptions

CDMException if problems with libxml or problems with input-file

15.93.2.2 virtual MetNoFimex::XMLDoc::~XMLDoc () [virtual]

15.93.3 Member Function Documentation

15.93.3.1 static boost::shared_ptr<XMLDoc> MetNoFimex::XMLDoc::fromFile (const std::string & *filename*) [static]

15.93.3.2 static boost::shared_ptr<XMLDoc> MetNoFimex::XMLDoc::fromString (const std::string & *buffer*, const std::string & *url* = "") [static]

15.93.3.3 static boost::shared_ptr<XMLDoc> MetNoFimex::XMLDoc::fromURL (const std::string & *url*) [static]

15.93.3.4 XPathObjPtr MetNoFimex::XMLDoc::getXPathObject (const std::string & *xpath*, xmlNodePtr *node* = 0) const

get a ptr to the node defined by xpath

Parameters

xpath xpath string for the node

Returns

an xpathobj, which is != 0, but might have 0 elements, i.e. nodesetval == 0 or nodesetval->nodeNr == 0

Exceptions

CDMException if xpath is not parsable

15.93.3.5 void MetNoFimex::XMLDoc::registerNamespace (const std::string & *prefix*, const std::string & *uri*)

register a namespace for later xpath

register a namespace with a prefix for later xpath retrievals

Parameters

prefix short name for namespace

uri full namespace name

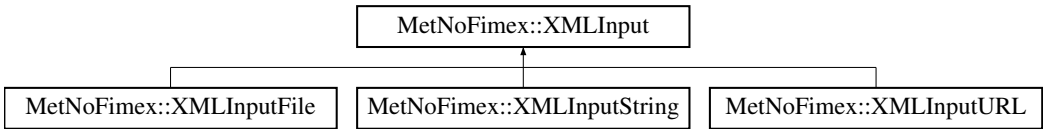
The documentation for this class was generated from the following file:

- include/fimex/XMLDoc.h

15.94 MetNoFimex::XMLInput Class Reference

`#include <XMLInput.h>`

Inheritance diagram for MetNoFimex::XMLInput:



Public Member Functions

- virtual ~XMLInput ()
- virtual boost::shared_ptr< XMLDoc > getXMLDoc () const =0
- virtual **std::string** id () const =0
- virtual bool isEmpty () const

15.94.1 Detailed Description

Interface for different XML sources like URL, file or string

15.94.2 Constructor & Destructor Documentation

15.94.2.1 virtual MetNoFimex::XMLInput::~XMLInput () [inline, virtual]

15.94.3 Member Function Documentation

15.94.3.1 virtual boost::shared_ptr<XMLDoc> MetNoFimex::XMLInput::getXMLDoc () const [pure virtual]

retrieve the XMLDoc

Returns

XMLDoc

Exceptions

CDMException

Implemented in MetNoFimex::XMLInputFile, MetNoFimex::XMLInputString, and MetNoFimex::XMLInputURL.

15.94.3.2 virtual std::string MetNoFimex::XMLInput::id () const [pure virtual]

return an identifier of the XMLInput

Implemented in MetNoFimex::XMLInputFile, MetNoFimex::XMLInputString, and MetNoFimex::XMLInputURL.

15.94.3.3 virtual bool MetNoFimex::XMLInput::isEmpty () const [inline, virtual]

check if information is available

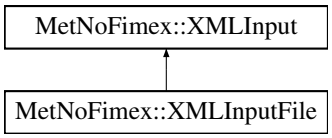
The documentation for this class was generated from the following file:

- include/fimex/XMLInput.h

15.95 MetNoFimex::XMLInputFile Class Reference

`#include <XMLInput.h>`

Inheritance diagram for MetNoFimex::XMLInputFile:



Public Member Functions

- XMLInputFile (const **std::string** &filename)
- virtual boost::shared_ptr< XMLDoc > getXMLDoc () const
- virtual **std::string** id () const

15.95.1 Constructor & Destructor Documentation

15.95.1.1 MetNoFimex::XMLInputFile::XMLInputFile (const std::string & *filename*) [inline]

15.95.2 Member Function Documentation

15.95.2.1 virtual boost::shared_ptr<XMLDoc> MetNoFimex::XMLInputFile::getXMLDoc () const [virtual]

retrieve the XMLDoc

Returns

XMLDoc

Exceptions

CDMException

Implements MetNoFimex::XMLInput.

15.95.2.2 virtual std::string MetNoFimex::XMLInputFile::id () const [inline, virtual]

return an identifier of the XMLInput

Implements MetNoFimex::XMLInput.

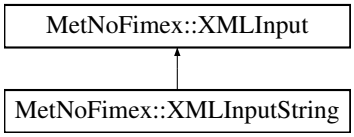
The documentation for this class was generated from the following file:

- include/fimex/XMLInput.h

15.96 MetNoFimex::XMLInputString Class Reference

`#include <XMLInput.h>`

Inheritance diagram for MetNoFimex::XMLInputString:



Public Member Functions

- XMLInputString (const **std::string** &content, const **std::string** &url="")
- virtual boost::shared_ptr< XMLDoc > getXMLDoc () const
- virtual **std::string** id () const

15.96.1 Constructor & Destructor Documentation

15.96.1.1 MetNoFimex::XMLInputString::XMLInputString (const std::string & *content*, const std::string & *url* = "") [inline]

parse a in-memory xml-content. The url will be neede for external references like style-definitions/-sheets or xincludes

15.96.2 Member Function Documentation

15.96.2.1 virtual boost::shared_ptr<XMLDoc> MetNoFimex::XMLInputString::getXMLDoc () const [virtual]

retrieve the XMLDoc

Returns

XMLDoc

Exceptions

CDMException

Implements MetNoFimex::XMLInput.

15.96.2.2 virtual std::string MetNoFimex::XMLInputString::id () const [inline, virtual]

return an identifier of the XMLInput

Implements MetNoFimex::XMLInput.

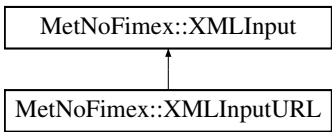
The documentation for this class was generated from the following file:

- include/fimex/XMLInput.h

15.97 MetNoFimex::XMLInputURL Class Reference

`#include <XMLInput.h>`

Inheritance diagram for MetNoFimex::XMLInputURL:



Public Member Functions

- XMLInputURL (const **std::string** &url)
- virtual boost::shared_ptr< XMLDoc > getXMLDoc () const
- virtual **std::string** id () const

15.97.1 Constructor & Destructor Documentation

15.97.1.1 MetNoFimex::XMLInputURL::XMLInputURL (const std::string & *url*) [inline]

15.97.2 Member Function Documentation

15.97.2.1 virtual boost::shared_ptr<XMLDoc> MetNoFimex::XMLInputURL::getXMLDoc () const [virtual]

retrieve the XMLDoc

Returns

XMLDoc

Exceptions

CDMException

Implements MetNoFimex::XMLInput.

15.97.2.2 virtual std::string MetNoFimex::XMLInputURL::id () const [inline, virtual]

return an identifier of the XMLInput

Implements MetNoFimex::XMLInput.

The documentation for this class was generated from the following file:

- include/fimex/XMLInput.h

Chapter 16

File Documentation

16.1 doxydoc.txt File Reference

16.2 include/felt/FeltConstants.h File Reference

```
#include <algorithm>
#include <boost/static_assert.hpp>
```

Namespaces

- namespace felt

Typedefs

- typedef short int felt::word

  A felt block "word" - 2 bytes.

Functions

- bool felt::isUndefined (word w)
- BOOST_STATIC_ASSERT (sizeof(felt::word)==2)

Variables

- const size_t felt::blockWords = 1024
- const size_t felt::blockSize = blockWords ∗ sizeof(word)
- const size_t felt::offsetToContentDefinition = 2
- const double felt::PI = 3.14159265358979323846264338327 95
- const double felt::EARTH_RADIUS = 6371000.

16.2.1 Function Documentation

16.2.1.1 BOOST_STATIC_ASSERT (sizeof(felt::word) = =2)

16.3 include/felt/FeltField.h File Reference

```
#include "FeltConstants.h"
#include <algorithm>
#include <boost/static_assert.hpp>
#include <string>
#include <vector>
#include <iosfwd>
#include <boost/array.hpp>
#include <boost/shared_array.hpp>
#include <boost/date_time/posix_time/posix_time_types.hpp>
#include <boost/noncopyable.hpp>
```

Classes

- class felt::FeltField

Namespaces

- namespace felt

Typedefs

- typedef boost::shared_ptr< FeltGridDefinition > felt::FeltGridDefinitionPtr

16.4 include/felt/FeltFile.h File Reference

```
#include "FeltConstants.h"
#include "FeltTypes.h"
#include <boost/shared_ptr.hpp>
#include <boost/weak_ptr.hpp>
#include <boost/filesystem/path.hpp>
#include <boost/date_time/posix_time/posix_time_types.hpp>
#include <boost/shared_array.hpp>
#include <boost/noncopyable.hpp>
#include <iterator>
#include <vector>
#include <iosfwd>
```

Classes

- class felt::FeltFile

Namespaces

- namespace felt

16.5 include/felt/FeltGridDefinition.h File Reference

```
#include <string>
#include <vector>
#include <iosfwd>
#include <boost/array.hpp>
```

Classes

- class felt::FeltGridDefinition

Namespaces

- namespace felt

Functions

- boost::array< float, 6 > felt::gridParameters (int gridType, int xNum, int yNum, int a, int b, int c, int d, const **std::vector**< short int > &extraData)
- **std::string** felt::gridParametersToProjDefinition (int gridType, const boost::array< float, 6 > &gridPars)
- **std::ostream** & felt::contentSummary (**std::ostream** &out, const FeltGridDefinition &grid)

16.6 include/felt/FeltTypeConversion.h File Reference

```
#include "FeltConstants.h"
#include <boost/date_time/posix_time/posix_time_types.hpp>
```

Namespaces

- namespace felt

Functions

- template<typename T >
  T felt::get (word w)
- boost::posix_time::ptime felt::parseTime (const word *data)
- boost::posix_time::ptime felt::parseTimeNoThrow (const word *data)

16.7 include/felt/FeltTypes.h File Reference

Namespaces

- namespace felt

16.8 include/fimex/binaryConstants.h File Reference

Classes

- struct binary< N >
- struct binary< 0 >

16.9 include/fimex/C_CDMReader.h File Reference

```
#include "fimex/CDMReader.h"
#include "fimex/c_fimex.h"
#include <map>
```

Classes

- class MetNoFimex::C_CDMReader

Namespaces

- namespace MetNoFimex

16.10 include/fimex/c_fimex.h File Reference

```
#include <stddef.h>
```

Typedefs

- typedef struct mifi_cdm_reader mifi_cdm_reader
- typedef int(∗ doubleDatasliceCallbackPtr)(mifi_cdm_reader ∗reader, const char ∗varName, size_t unLimDimPos, double ∗scaledData, size_t dataSize)

Functions

- void mifi_free_cdm_reader (mifi_cdm_reader ∗reader)
- mifi_cdm_reader ∗ mifi_new_felt_reader (const char ∗filename, const char ∗configFile)
- mifi_cdm_reader ∗ mifi_new_netcdf_reader (const char ∗filename)
- mifi_cdm_reader ∗ mifi_new_grib_reader (const char ∗filename, const char ∗configFile)
- mifi_cdm_reader ∗ mifi_new_ncml_reader (const char ∗ncmlFile)
- mifi_cdm_reader ∗ mifi_new_ncml_modifier (mifi_cdm_reader ∗reader, const char ∗ncmlFile)
- int mifi_netcdf_writer (mifi_cdm_reader ∗reader, const char ∗filename, const char ∗configFile, int version)
- int mifi_grib_writer (mifi_cdm_reader ∗reader, const char ∗filename, const char ∗configFile, int version)
- int mifi_nullcdm_writer (mifi_cdm_reader ∗reader)
- mifi_cdm_reader ∗ mifi_new_cdminterpolator (mifi_cdm_reader ∗reader, int method, const char ∗proj_input, const char ∗out_x_axis, const char ∗out_y_axis, const char ∗out_x_axis_unit, const char ∗out_y_axis_unit)

  change the projection of the reader to this new projection

- mifi_cdm_reader ∗ mifi_new_c_reader (mifi_cdm_reader ∗reader)
- int mifi_set_callback_double (mifi_cdm_reader ∗c_reader, const char ∗varName, doubleDataslice-CallbackPtr callback)
- size_t mifi_get_variable_number (mifi_cdm_reader ∗reader)
- const char ∗ mifi_get_variable_name (mifi_cdm_reader ∗reader, size_t pos)
- int mifi_get_double_dataslize (mifi_cdm_reader ∗reader, const char ∗varName, size_t unLimDim-Pos, double ∗∗data, size_t ∗size)
- int mifi_get_double_data (mifi_cdm_reader ∗reader, const char ∗varName, double ∗∗data, size_t ∗size)

16.10.1 Typedef Documentation

16.10.1.1 typedef int(∗ doubleDatasliceCallbackPtr)(mifi_cdm_reader ∗reader, const char ∗varName, size_t unLimDimPos, double ∗scaledData, size_t dataSize)

Function pointer as used for the get_double_dataslice callback function

Returns

0 on success, error otherwise

16.10.1.2 typedef struct mifi_cdm_reader mifi_cdm_reader

This is the public C-Api for fimex. It is a wrapper api for the underlying C++ api.

16.10.2 Function Documentation

16.10.2.1 void mifi_free_cdm_reader (mifi_cdm_reader * *reader*)

Free the reader. This won't free the resources immediately, but reduce the reference counter. It is therefore possible to free a reader, while it still is used within another part of the fimex-chain.

16.10.2.2 int mifi_get_double_data (mifi_cdm_reader * *reader*, const char * *varName*, double ** *data*, size_t * *size*)

get all the data from the dataReader

Parameters

reader dataReader to read the data from

varName variable name associated with the data

data,: the returned data. It will be allocated automatically, it is the task of the user to **free** it. Undefined values will be NaN.

size,: the size of the returned data.

Returns

0 on success

16.10.2.3 int mifi_get_double_dataslize (mifi_cdm_reader * *reader*, const char * *varName*, size_t *unLimDimPos*, double ** *data*, size_t * *size*)

get a slice of data from the dataReader

Parameters

reader dataReader to read the data from

varName variable name associated with the data

unLimDimPos unlimited dimension of the slice

data,: the returned data. It will be allocated automatically, it is the task of the user to **free** it. Undefined values will be NaN.

size,: the size of the returned data.

Returns

0 on success

16.10.2.4 const char* mifi_get_variable_name (mifi_cdm_reader * *reader*, size_t *pos*)

Get the name of a variable from the reader.

Parameters

reader the data source

pos the position number of the variable, should be between 0 and size-1

Returns

the variable name, or NULL on failure

16.10.2.5 size_t mifi_get_variable_number (mifi_cdm_reader * *reader*)

Get the number of the variables from the reader.

Parameters

reader the data source

Returns

the number of variables

16.10.2.6 int mifi_grib_writer (mifi_cdm_reader * *reader*, const char * *filename*, const char * *configFile*, int *version*)

Write the content of the reader to the filename as gribfile.

Parameters

reader the data source

filename the name of the grib-file to write

configFile an optional configFile, use "" or 0 if not needed

version,the version of the grib-edition. Implemented are 1 or 2.

Returns

0 on success.

16.10.2.7 int mifi_netcdf_writer (mifi_cdm_reader * *reader*, const char * *filename*, const char * *configFile*, int *version*)

Write the content of the reader to the filename.

Parameters

reader the data source

filename the name of the netcdf-file to write

configFile an optional configFile, use "" or 0 if not needed

version,the version of the netcdf-file. Implemented are 3 or 4.

Returns

0 on success.

16.10.2.8 mifi_cdm_reader* mifi_new_c_reader (mifi_cdm_reader * *reader*)

Get a new reader which allows setting c-callback functions.

Parameters

the original data-source

Returns

the reader object-pointer, use mifi_freeCDMReader to free, or NULL on error.

16.10.2.9 mifi_cdm_reader* mifi_new_cdminterpolator (mifi_cdm_reader * *reader*, int *method*, const char * *proj_input*, const char * *out_x_axis*, const char * *out_y_axis*, const char * *out_x_axis_unit*, const char * *out_y_axis_unit*)

change the projection of the reader to this new projection

Parameters

method Interpolation method

proj_input input-string for proj4, used as output projection

out_x_axis config-string for x_axis, either '1,2,...,5' or 'auto' or 'auto,distance=3.5'

out_y_axis config-string for y_axis, either '1,2,...,5' or 'auto' or 'auto,distance=3.5'

out_x_axis_unit unit of the output x-axis

out_y_axis_unit unit of the output y-axis

Returns

the reader object-pointer, use mifi_freeCDMReader to free, or NULL on error.

16.10.2.10 mifi_cdm_reader* mifi_new_felt_reader (const char * *filename*, const char * *configFile*)

Get a new reader from a felt file.

Parameters

filename name of the felt-file

configFile configuration file for the felt-file

Returns

the reader object-pointer, use mifi_freeCDMReader to free, or NULL on error.

16.10.2.11 mifi_cdm_reader* mifi_new_grib_reader (const char * *filename*, const char * *configFile*)

Get a new reader from a grib1/2 file.

Parameters

filename name of the grib-file

configFile configuration file for the grib-file

Returns

the reader object-pointer, use mifi_freeCDMReader to free, or NULL on error.

16.10.2.12 mifi_cdm_reader* mifi_new_ncml_modifier (mifi_cdm_reader * *reader*, const char * *ncmlFile*)

Modify a reader using a ncml file.

Parameters

reader the data/cdm source
ncmlFile name of the ncml config file

Returns

the reader object-pointer, use mifi_freeCDMReader to free, or NULL on error.

16.10.2.13 mifi_cdm_reader* mifi_new_ncml_reader (const char * *ncmlFile*)

Get a new reader from a ncml file.

Parameters

ncmlFile name of the ncml config file

Returns

the reader object-pointer, use mifi_freeCDMReader to free, or NULL on error.

16.10.2.14 mifi_cdm_reader* mifi_new_netcdf_reader (const char * *filename*)

Get a new reader from a netcdf file.

Parameters

filename name of the felt-file
configFile configuration file for the felt-file

Returns

the reader object-pointer, use mifi_freeCDMReader to free, or NULL on error.

16.10.2.15 int mifi_nullcdm_writer (mifi_cdm_reader * *reader*)

Fetch the whole data belonging to the cdm, but don't write it anywhere.

Parameters

reader the data source

Returns

0 on success.

16.10.2.16 int mifi_set_callback_double (mifi_cdm_reader * *c_reader*, const char * *varName*, doubleDatasliceCallbackPtr *callback*)

Add a callback for a variable. The variable will be converted to datatype double.

Parameters

c_reader the reader as created by mifi_new_c_reader

varName the name of the variable

callback a function-ptr to the callback function

Returns

0 on success, else error

Warning

the callback function will only be able to modify data which is available in the reader. It cannot change any information the writer request, but the reader doesn't now about. This data will continue to be undefined!

16.11 include/fimex/CachedForwardInterpolation.h File Reference

```
#include "fimex/CachedInterpolation.h"
#include <boost/shared_array.hpp>
#include <vector>
```

Classes

- class MetNoFimex::CachedForwardInterpolation

Namespaces

- namespace MetNoFimex

16.12 include/fimex/CachedInterpolation.h File Reference

```
#include <boost/shared_array.hpp>
#include "fimex/interpolation.h"
#include "fimex/Data.h"
```

Classes

- class MetNoFimex::CachedInterpolationInterface
- class MetNoFimex::CachedInterpolation

Namespaces

- namespace MetNoFimex

16.13 include/fimex/CachedVectorReprojection.h File Reference

```
#include <boost/shared_ptr.hpp>
#include "fimex/Data.h"
#include "fimex/interpolation.h"
```

Classes

- class MetNoFimex::CachedVectorReprojection

Namespaces

- namespace MetNoFimex

16.14 include/fimex/CDM.h File Reference

```
#include <map>
#include <vector>
#include <string>
#include <ostream>
#include <boost/regex_fwd.hpp>
#include "fimex/CDMAttribute.h"
#include "fimex/CDMVariable.h"
#include "fimex/CDMDimension.h"
#include "fimex/CDMException.h"
#include "fimex/CDMconstants.h"
#include "fimex/coordSys/Projection.h"
```

Classes

- class MetNoFimex::CDM

  Data structure of the Common Data Model.

Namespaces

- namespace MetNoFimex

16.15 include/fimex/CDMAttribute.h File Reference

```
#include <string>
#include <vector>
#include <ostream>
#include <boost/shared_ptr.hpp>
#include "fimex/CDMDataType.h"
#include "fimex/CDMNamedEntity.h"
#include "fimex/CDMException.h"
#include "fimex/deprecated.h"
```

Classes

- class MetNoFimex::CDMAttribute

Namespaces

- namespace MetNoFimex

Functions

- MetNoFimex::DEPRECATED (**std::vector**< CDMAttribute > projStringToAttributes(**std::string** projStr))

  convert a proj4 string to a list of CDMAttributes usable for CF-1.0 projection variable

- MetNoFimex::DEPRECATED (**std::string** attributesToProjString(const **std::vector**< CDMAttribute > &attrs))

  convert attributes of a projection-variable to a projString

16.16 include/fimex/CDMconstants.h File Reference

```
#include "fimex/deprecated.h"
```

Defines

- #define MIFI_EARTH_RADIUS_M 6371000

  constants used through-out fimex

- #define MIFI_FILETYPE_UNKNOWN -1
- #define MIFI_FILETYPE_FELT 0
- #define MIFI_FILETYPE_NETCDF 1
- #define MIFI_FILETYPE_NCML 2
- #define MIFI_FILETYPE_GRIB 3
- #define MIFI_FILETYPE_WDB 4
- #define MIFI_FILETYPE_METGM 5

Functions

- const char * fimexVersion ()
- int mifi_get_filetype (const char *filetypeName)

  get the filetype of a filetype name

- const char * mifi_get_filetype_name (int filetype)

  get the filetype-name of a filetype

- int mifi_get_max_filetype_number ()
- int fimexHas (int fileType)
- DEPRECATED (int fimexHasNetcdf())

16.16.1 Define Documentation

16.16.1.1 #define MIFI_EARTH_RADIUS_M 6371000

constants used through-out fimex

CDMConstants stores several constants used in fimex, accessible from C and C++. Constants are either available as macro, or as function. the default radius of a sperical earth in meter

16.16.1.2 #define MIFI_FILETYPE_FELT 0

16.16.1.3 #define MIFI_FILETYPE_GRIB 3

16.16.1.4 #define MIFI_FILETYPE_METGM 5

16.16.1.5 #define MIFI_FILETYPE_NCML 2

16.16.1.6 #define MIFI_FILETYPE_NETCDF 1

16.16.1.7 #define MIFI_FILETYPE_UNKNOWN -1

The MIFI_FILETYPE_∗ define the available input and output file-formats

16.16.1.8 #define MIFI_FILETYPE_WDB 4

16.16.2 Function Documentation

16.16.2.1 DEPRECATED (int *fimexHasNetcdf*())

check if fimex is configured with netcdf-support

Deprecated

use fimexHas(fileType)

check if fimex is configured with grib_api-support

Deprecated

use fimexHas(fileType)

check if fimex is configured with felt-support

Deprecated

use fimexHas(fileType)

16.16.2.2 int fimexHas (int *fileType*)

check if fimex is configured with the filetype

Parameters

fileType one of the MIFI_FILETYPE_∗ define constants

16.16.2.3 const char∗ fimexVersion ()

version of fimex

16.16.2.4 int mifi_get_filetype (const char * *filetypeName*)

get the filetype of a filetype name

Returns

one of MIFI_FILETYPE_*

16.16.2.5 const char* mifi_get_filetype_name (int *filetype*)

get the filetype-name of a filetype

Parameters

one of MIFI_FILETYPE_*

16.16.2.6 int mifi_get_max_filetype_number ()

get the maximum number of filetypes, that is , the largest number of valid filetype you can get.

16.17 include/fimex/CDMDataType.h File Reference

```
#include <string>
```

Namespaces

- namespace MetNoFimex

Enumerations

- enum MetNoFimex::CDMDataType {
  MetNoFimex::CDM_NAT = 0, MetNoFimex::CDM_CHAR, MetNoFimex::CDM_SHORT, MetNoFimex::CDM_INT,
  MetNoFimex::CDM_FLOAT, MetNoFimex::CDM_DOUBLE, MetNoFimex::CDM_STRING, MetNoFimex::CDM_UCHAR,
  MetNoFimex::CDM_USHORT, MetNoFimex::CDM_UINT, MetNoFimex::CDM_INT64, MetNoFimex::CDM_UINT64 }

Functions

- CDMDataType MetNoFimex::string2datatype (const **std::string** &s)
  translate float/string/... to the appropriate CDMDataType
- **std::string** MetNoFimex::datatype2string (CDMDataType type)

16.18 include/fimex/CDMDimension.h File Reference

```
#include <string>
#include <ostream>
#include "fimex/CDMNamedEntity.h"
```

Classes

- class MetNoFimex::CDMDimension

Namespaces

- namespace MetNoFimex

16.19 include/fimex/CDMException.h File Reference

```
#include <stdexcept>
#include <string>
```

Classes

- class MetNoFimex::CDMException

Namespaces

- namespace MetNoFimex

16.20 include/fimex/CDMExtractor.h File Reference

```
#include <map>
#include <set>
#include <boost/array.hpp>
#include <boost/shared_ptr.hpp>
#include "fimex/CDMReader.h"
#include "fimex/CDMDataType.h"
#include "fimex/TimeUnit.h"
#include "fimex/coordSys/CoordinateAxis.h"
```

Classes

- class MetNoFimex::CDMExtractor

Namespaces

- namespace MetNoFimex

16.21 include/fimex/CDMFileReaderFactory.h File Reference

```
#include <boost/shared_ptr.hpp>
#include <vector>
#include <string>
#include "fimex/XMLInput.h"
#include "fimex/deprecated.h"
```

Classes

- class MetNoFimex::CDMFileReaderFactory

Namespaces

- namespace MetNoFimex

16.22 include/fimex/CDMInterpolator.h File Reference

```
#include <vector>
#include "fimex/CDMReader.h"
#include "fimex/CachedInterpolation.h"
#include "fimex/CachedVectorReprojection.h"
#include "fimex/deprecated.h"
```

Classes

- class MetNoFimex::InterpolatorProcess2d
- class MetNoFimex::InterpolatorFill2d
- class MetNoFimex::InterpolatorCreepFill2d
- class MetNoFimex::CDMInterpolator

Namespaces

- namespace MetNoFimex

16.23 include/fimex/CDMNamedEntity.h File Reference

```
#include <string>
#include <functional>
#include <boost/shared_ptr.hpp>
```

Classes

- class MetNoFimex::CDMNamedEntity
- struct MetNoFimex::CDMNameCompare
- class MetNoFimex::CDMNameEqual
- class MetNoFimex::CDMNameEqualPtr

Namespaces

- namespace MetNoFimex

16.24 include/fimex/CDMPressureConversions.h File Reference

```
#include "fimex/CDMReader.h"
```

Classes

- class MetNoFimex::CDMPressureConversions

Namespaces

- namespace MetNoFimex

16.25 include/fimex/CDMQualityExtractor.h File Reference

```
#include "CDMReader.h"
#include <boost/shared_ptr.hpp>
#include <boost/noncopyable.hpp>
#include "fimex/CDMException.h"
#include "fimex/SliceBuilder.h"
#include <vector>
#include <map>
```

Classes

- class MetNoFimex::CDMQualityExtractor

  Extract data with defined quality status.

Namespaces

- namespace MetNoFimex

16.26 include/fimex/CDMReader.h File Reference

```
#include <boost/shared_ptr.hpp>
#include <boost/noncopyable.hpp>
#include "fimex/CDMException.h"
#include "fimex/SliceBuilder.h"
```

Classes

- class MetNoFimex::CDMReader

  Basic interface for CDM reading and manipulation classes.

Namespaces

- namespace MetNoFimex

16.27 include/fimex/CDMReaderUtils.h File Reference

```
#include "fimex/CDMReader.h"
#include <boost/date_time/posix_time/posix_time_types.hpp>
#include <boost/shared_ptr.hpp>
```

Namespaces

- namespace MetNoFimex

Functions

- boost::posix_time::ptime MetNoFimex::getUniqueForecastReferenceTime (boost::shared_ptr< CDMReader > reader)

16.28 include/fimex/CDMTimeInterpolator.h File Reference

```
#include "CDMReader.h"
#include <map>
#include <vector>
```

Classes

- class MetNoFimex::CDMTimeInterpolator

Namespaces

- namespace MetNoFimex

16.29 include/fimex/CDMVariable.h File Reference

```
#include <string>
#include <vector>
#include <ostream>
#include "fimex/CDMAttribute.h"
#include "fimex/CDMDataType.h"
#include "fimex/CDMNamedEntity.h"
```

Classes

- class MetNoFimex::CDMVariable

Namespaces

- namespace MetNoFimex

16.30 include/fimex/CDMVerticalInterpolator.h File Reference

```
#include <vector>
#include "fimex/mifi_constants.h"
#include "fimex/CDMReader.h"
#include "fimex/coordSys/CoordinateSystem.h"
```

Classes

- class MetNoFimex::CDMVerticalInterpolator

  Interpolation of vertical layers.

Namespaces

- namespace MetNoFimex

16.31 include/fimex/CDMWriter.h File Reference

```
#include <string>
#include <boost/shared_ptr.hpp>
#include "fimex/CDMReader.h"
```

Classes

- class MetNoFimex::CDMWriter

Namespaces

- namespace MetNoFimex

16.32 include/fimex/CoordinateSystemSliceBuilder.h File Reference

```
#include "SliceBuilder.h"
#include <string>
#include <vector>
#include <map>
#include <set>
#include <boost/shared_ptr.hpp>
```

Classes

- class MetNoFimex::CoordinateSystemSliceBuilder

Namespaces

- namespace MetNoFimex

16.33 include/fimex/coordSys/AlbersConicalEqualAreaProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
```

Classes

- class MetNoFimex::AlbersConicalEqualAreaProjection

Namespaces

- namespace MetNoFimex

16.34 include/fimex/coordSys/AzimuthalEquidistantProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
```

Classes

- class MetNoFimex::AzimuthalEquidistantProjection

Namespaces

- namespace MetNoFimex

16.35 include/fimex/coordSys/CoordinateAxis.h File Reference

```
#include "fimex/CDMVariable.h"
```

Classes

- class MetNoFimex::CoordinateAxis

Namespaces

- namespace MetNoFimex

Functions

- **std::ostream** & MetNoFimex::operator<< (**std::ostream** &out, CoordinateAxis ca)
- **std::ostream** & MetNoFimex::operator<< (**std::ostream** &out, CoordinateAxis::AxisType t)

16.36 include/fimex/coordSys/CoordinateSystem.h File Reference

```
#include <functional>
#include <vector>
#include <boost/shared_ptr.hpp>
#include <iostream>
#include "fimex/coordSys/CoordinateAxis.h"
#include "fimex/coordSys/Projection.h"
```

Classes

- class MetNoFimex::CoordinateSystem
- struct MetNoFimex::CompleteCoordinateSystemForComparator

Namespaces

- namespace MetNoFimex

Functions

- **std::ostream** & MetNoFimex::operator<< (**std::ostream** &out, const CoordinateSystem &p)
- **std::vector**< boost::shared_ptr< const CoordinateSystem > > MetNoFimex::listCoordinateSystems (const CDM &cdm)

16.37 include/fimex/coordSys/LambertAzimuthalEqualAreaProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
```

Classes

- class MetNoFimex::LambertAzimuthalEqualAreaProjection

Namespaces

- namespace MetNoFimex

16.38 include/fimex/coordSys/LambertConformalConicProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
#include "fimex/Data.h"
```

Classes

- class MetNoFimex::LambertConformalConicProjection

Namespaces

- namespace MetNoFimex

16.39 include/fimex/coordSys/LambertCylindricalEqualAreaProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
```

Classes

- class MetNoFimex::LambertCylindricalEqualAreaProjection

Namespaces

- namespace MetNoFimex

16.40 include/fimex/coordSys/LatitudeLongitudeProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
```

Classes

- class MetNoFimex::LatitudeLongitudeProjection

Namespaces

- namespace MetNoFimex

16.41 include/fimex/coordSys/MercatorProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
```

Classes

- class MetNoFimex::MercatorProjection

Namespaces

- namespace MetNoFimex

16.42 include/fimex/coordSys/OrthographicProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
```

Classes

- class MetNoFimex::OrthographicProjection

Namespaces

- namespace MetNoFimex

16.43 include/fimex/coordSys/PolarStereographicProjection.h File Reference

```
#include "fimex/coordSys/StereographicProjection.h"
```

Classes

- class MetNoFimex::PolarStereographicProjection

Namespaces

- namespace MetNoFimex

16.44 include/fimex/coordSys/Projection.h File Reference

```
#include <vector>
#include <iostream>
#include <boost/shared_ptr.hpp>
#include "fimex/CDMAttribute.h"
```

Classes

- class MetNoFimex::Projection

Namespaces

- namespace MetNoFimex

Functions

- **std::ostream** & MetNoFimex::operator<< (**std::ostream** &out, const Projection &proj)

16.45 include/fimex/coordSys/ProjectionImpl.h File Reference

```
#include "fimex/coordSys/Projection.h"
```

Classes

- class MetNoFimex::ProjectionImpl

Namespaces

- namespace MetNoFimex

16.46 include/fimex/coordSys/RotatedLatitudeLongitudeProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
#include "fimex/Data.h"
```

Classes

- class MetNoFimex::RotatedLatitudeLongitudeProjection

Namespaces

- namespace MetNoFimex

16.47 include/fimex/coordSys/StereographicProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
```

Classes

- class MetNoFimex::StereographicProjection

Namespaces

- namespace MetNoFimex

16.48 include/fimex/coordSys/TransverseMercatorProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
```

Classes

- class MetNoFimex::TransverseMercatorProjection

Namespaces

- namespace MetNoFimex

16.49 include/fimex/coordSys/UnknownToFgdcProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
```

Classes

- class MetNoFimex::UnknownToFgdcProjection

Namespaces

- namespace MetNoFimex

16.50 include/fimex/coordSys/VerticalPerspectiveProjection.h File Reference

```
#include "fimex/coordSys/ProjectionImpl.h"
```

Classes

- class MetNoFimex::VerticalPerspectiveProjection

Namespaces

- namespace MetNoFimex

16.51 include/fimex/Data.h File Reference

```
#include <boost/shared_array.hpp>
#include <boost/shared_ptr.hpp>
#include <string>
#include <sstream>
#include <iostream>
#include "fimex/CDMDataType.h"
#include "fimex/CDMException.h"
#include "fimex/Utils.h"
```

Classes

- class MetNoFimex::Data

Namespaces

- namespace MetNoFimex

Functions

- boost::shared_ptr< Data > MetNoFimex::createData (CDMDataType datatype, size_t length, double val=0)

  create a Data-pointer of the datatype

- boost::shared_ptr< Data > MetNoFimex::createData (size_t length, boost::shared_array< double > array)

  create a Data-pointer of type CDM_DOUBLE

- boost::shared_ptr< Data > MetNoFimex::createData (size_t length, boost::shared_array< float > array)

  create a Data-pointer of type CDM_FLOAT

- boost::shared_ptr< Data > MetNoFimex::createData (size_t length, boost::shared_array< int > array)

  create a Data-pointer of type CDM_INT

- boost::shared_ptr< Data > MetNoFimex::createData (size_t length, boost::shared_array< short > array)

  create a Data-pointer of type CDM_SHORT

- boost::shared_ptr< Data > MetNoFimex::createData (size_t length, boost::shared_array< char > array)

  create a Data-pointer of type CDM_CHAR

- boost::shared_ptr< Data > MetNoFimex::createData (size_t length, boost::shared_array< unsigned int > array)

create a Data-pointer of type CDM_UINT

- boost::shared_ptr< Data > MetNoFimex::createData (size_t length, boost::shared_array< long long > array)

  create a Data-pointer of type CDM_INT64

- boost::shared_ptr< Data > MetNoFimex::createData (size_t length, boost::shared_array< unsigned long long > array)

  create a Data-pointer of type CDM_UINT64

- boost::shared_ptr< Data > MetNoFimex::createData (size_t length, boost::shared_array< unsigned short > array)

  create a Data-pointer of type CDM_USHORT

- boost::shared_ptr< Data > MetNoFimex::createData (size_t length, boost::shared_array< unsigned char > array)

  create a Data-pointer of type CDM_UCHAR

- template<class InputIterator >
  boost::shared_ptr< Data > MetNoFimex::createData (CDMDataType datatype, InputIterator first, InputIterator last)

  create a Data-pointer of the datatype and fill with the data from the iterator

- boost::shared_ptr< Data > MetNoFimex::createDataSlice (CDMDataType datatype, const Data &data, size_t dataStartPos, size_t dataSize)

  create a one-dimensional dataslice from another Data object

16.52 include/fimex/DataTypeChanger.h File Reference

```
#include "fimex/CDMDataType.h"
#include "boost/shared_ptr.hpp"
#include "fimex/CDMException.h"
```

Classes

- class MetNoFimex::DataTypeChanger

Namespaces

- namespace MetNoFimex

16.53 include/fimex/deprecated.h File Reference

Defines

- #define DEPRECATED(func) func

16.53.1 Define Documentation

16.53.1.1 #define DEPRECATED(func) func

16.54 include/fimex/Felt_Array2.h File Reference

```
#include <string>
#include <set>
#include <vector>
#include <map>
#include <boost/array.hpp>
#include <boost/date_time/posix_time/posix_time_types.hpp>
#include <fimex/Felt_Types.h>
#include "fimex/Felt_File_Error.h"
#include "felt/FeltTypes.h"
```

Classes

- class MetNoFelt::Felt_Array2

  A collection of FeltFields to build a 4-dimensional data array for 1-parameter and 1-vertical coordinate.

Namespaces

- namespace MetNoFelt

16.55 include/fimex/Felt_File2.h File Reference

```
#include <map>
#include <vector>
#include <string>
#include <boost/shared_ptr.hpp>
#include <boost/shared_array.hpp>
#include "fimex/Data.h"
#include "fimex/Felt_Types.h"
#include "fimex/Felt_File_Error.h"
#include "fimex/FeltParameters.h"
#include "fimex/Logger.h"
#include "felt/FeltTypes.h"
```

Classes

- class MetNoFelt::Felt_File2

  Felt File access.

Namespaces

- namespace MetNoFelt

16.56 include/fimex/Felt_File_Error.h File Reference

```
#include <stdexcept>
#include <string>
```

Classes

- class MetNoFelt::Felt_File_Error

Namespaces

- namespace MetNoFelt

16.57 include/fimex/Felt_Types.h File Reference

Classes

- struct MetNoFelt::LevelPairLess

Namespaces

- namespace MetNoFelt

Typedefs

- typedef **std::pair**< short, short > MetNoFelt::LevelPair

16.58 include/fimex/FeltCDMReader2.h File Reference

```
#include <string>
#include <vector>
#include <map>
#include <boost/shared_ptr.hpp>
#include "fimex/CDMReader.h"
#include "fimex/CDMDimension.h"
#include "fimex/Felt_Types.h"
#include "fimex/ReplaceStringObject.h"
#include "fimex/XMLInput.h"
#include <boost/date_time/posix_time/posix_time_types.hpp>
```

Classes

- class MetNoFimex::FeltCDMReader2

Namespaces

- namespace MetNoFelt
- namespace MetNoFimex

16.59 include/fimex/FeltParameters.h File Reference

```
#include <map>
#include <string>
#include <vector>
#include <boost/array.hpp>
#include "fimex/Felt_File_Error.h"
```

Classes

- class MetNoFelt::FeltParameters

Namespaces

- namespace MetNoFelt

Functions

- **std::string** MetNoFelt::getProjString (int gridType, const boost::array< float, 6 > &gridParameters)
- const int MetNoFelt::ANY_VALUE ()
- const **std::string** & MetNoFelt::UNDEFINED ()
- const boost::array< short, 16 > & MetNoFelt::ANY_ARRAY ()
- const boost::array< short, 20 > & MetNoFelt::ANY_ARRAY20 ()

16.60 include/fimex/GribApiCDMWriter.h File Reference

```
#include "fimex/CDMWriter.h"
```

Classes

- class MetNoFimex::GribApiCDMWriter

Namespaces

- namespace MetNoFimex

16.61 include/fimex/GribApiCDMWriter_Impl1.h File Reference

```
#include "fimex/GribApiCDMWriter_ImplAbstract.h"
```

Classes

- class MetNoFimex::GribApiCDMWriter_Impl1

Namespaces

- namespace MetNoFimex

16.62 include/fimex/GribApiCDMWriter_Impl2.h File Reference

```
#include "fimex/GribApiCDMWriter_ImplAbstract.h"
```

Classes

- class MetNoFimex::GribApiCDMWriter_Impl2

Namespaces

- namespace MetNoFimex

16.63 include/fimex/GribApiCDMWriter_ImplAbstract.h File Reference

```
#include <vector>
#include <fstream>
#include <iostream>
#include "fimex/Logger.h"
#include "fimex/CDMWriter.h"
#include "fimex/XMLDoc.h"
#include "fimex/CDMException.h"
#include "fimex/TimeUnit.h"
```

Classes

- class MetNoFimex::GribApiCDMWriter_ImplAbstract

Namespaces

- namespace MetNoFimex

16.64 include/fimex/GribCDMReader.h File Reference

```
#include <vector>
#include <map>
#include <set>
#include "boost/shared_ptr.hpp"
#include "fimex/GribFileIndex.h"
#include "fimex/CDMReader.h"
#include "fimex/ReplaceStringObject.h"
#include "fimex/XMLInput.h"
```

Classes

- class MetNoFimex::GribCDMReader

Namespaces

- namespace MetNoFimex

16.65 include/fimex/GribFileIndex.h File Reference

```
#include <boost/date_time/posix_time/posix_time.hpp>
#include <boost/filesystem/operations.hpp>
#include <vector>
#include "fimex/XMLDoc.h"
#include "fimex/GridDefinition.h"
#include <boost/date_time/posix_time/posix_time_types.hpp>
```

Classes

- class MetNoFimex::GribFileMessage
- class MetNoFimex::GribFileMessageEqualTime

  Functor to find Messages with equal time.

- class MetNoFimex::GribFileMessageEqualLevelTime

  Functor to find messages with equal level and time.

- class MetNoFimex::GribFileIndex

Namespaces

- namespace MetNoFimex

Functions

- **std::ostream** & MetNoFimex::operator<< (**std::ostream** &os, const GribFileMessage &gfm)

  outputstream for a GribFileMessage

- **std::ostream** & MetNoFimex::operator<< (**std::ostream** &os, const GribFileIndex &gfm)

  outputstream for a GribFileIndex

16.66 include/fimex/GribUtils.h File Reference

```
#include "fimex/GridDefinition.h"
#include <stdexcept>
```

Namespaces

- namespace MetNoFimex

Defines

- #define MIFI_GRIB_CHECK(error, msg) mifi_grib_check(error, msg, __LINE__, __FILE__);

Functions

- void mifi_grib_check (int error, const char *msg, int line, const char *file) throw (std::runtime_error)
- GridDefinition::Orientation MetNoFimex::gribGetGridOrientation (boost::shared_ptr< grib_handle > gh)

16.66.1 Define Documentation

16.66.1.1 #define MIFI_GRIB_CHECK(error, msg) mifi_grib_check(error, msg, __LINE__, __FILE__);

macro to call mifi_grib_check with correct line and file

16.66.2 Function Documentation

16.66.2.1 void mifi_grib_check (int *error*, const char * *msg*, int *line*, const char * *file*) throw (std::runtime_error)

runtime-exception checker for grib_check

16.67 include/fimex/GridDefinition.h File Reference

```
#include "fimex/binaryConstants.h"
#include <string>
#include "boost/shared_ptr.hpp"
```

Classes

- class MetNoFimex::GridDefinition

Namespaces

- namespace MetNoFimex

16.68 include/fimex/interpolation.h File Reference

```
#include "fimex/mifi_constants.h"
```

Functions

- int mifi_interpolate_f (int method, const char *proj_input, const float *infield, const double *in_x_axis, const double *in_y_axis, const int in_x_axis_type, const int in_y_axis_type, const int ix, const int iy, const int iz, const char *proj_output, float *outfield, const double *out_x_axis, const double *out_y_axis, const int out_x_axis_type, const int out_y_axis_type, const int ox, const int oy)
- int mifi_interpolate_d (int method, char *proj_input, double *infield, double *in_x_axis, double *in_y_axis, int in_x_axis_type, int in_y_axis_type, int ix, int iy, int iz, char *proj_output, double *outfield, double *out_x_axis, double *out_y_axis, int out_x_axis_type, int out_y_axis_type, int ox, int oy)

  not implemented yet

- int mifi_vector_reproject_values_f (int method, const char *proj_input, const char *proj_output, float *u_out, float *v_out, const double *out_x_axis, const double *out_y_axis, int out_x_axis_type, int out_y_axis_type, int ox, int oy, int oz)

  interpolate the vector values

- int mifi_vector_reproject_values_by_matrix_f (int method, const double *matrix, float *u_out, float *v_out, int ox, int oy, int oz)
- int mifi_get_vector_reproject_matrix (const char *proj_input, const char *proj_output, const double *out_x_axis, const double *out_y_axis, int out_x_axis_type, int out_y_axis_type, int ox, int oy, double *matrix)
- int mifi_get_values_f (const float *infield, float *outfield, const double x, const double y, const int ix, const int iy, const int iz)
- int mifi_get_values_bilinear_f (const float *infield, float *outvalues, const double x, const double y, const int ix, const int iy, const int iz)
- int mifi_get_values_bicubic_f (const float *infield, float *outvalues, const double x, const double y, const int ix, const int iy, const int iz)

  not implemented yet

- int mifi_get_values_linear_f (const float *infieldA, const float *infieldB, float *outfield, const size_t n, const double a, const double b, const double x)
- int mifi_get_values_linear_d (const double *infieldA, const double *infieldB, double *outfield, const size_t n, const double a, const double b, const double x)
- int mifi_get_values_log_f (const float *infieldA, const float *infieldB, float *outfield, const size_t n, const double a, const double b, const double x)
- int mifi_get_values_log_log_f (const float *infieldA, const float *infieldB, float *outfield, const size_t n, const double a, const double b, const double x)
- int mifi_points2position (double *points, const int n, const double *axis, const int num, const int axis_type)

  find position in array of position in projection

- int mifi_3d_array_position (int x, int y, int z, int ix, int iy, int iz)
- int mifi_project_values (const char *proj_input, const char *proj_output, double *in_out_x_vals, double *in_out_y_vals, const int num)

  project values so that the projetion (x,y) => (x_proj), (y_proj) can be expressed as x_proj(x,y), y_proj(x,y)

- int mifi_project_axes (const char *proj_input, const char *proj_output, const double *in_x_axis, const double *in_y_axis, const int ix, const int iy, double *out_xproj_axis, double *out_yproj_axis)

  project axes so that the projetion (x,y) => (x_proj), (y_proj) can be expressed as x_proj(x,y), y_proj(x,y)

- int mifi_fill2d_f (size_t nx, size_t ny, float *field, float relaxCrit, float corrEff, size_t maxLoop, size_t *nChanged)

  Method to fill undefined values in a 2d field.

- int mifi_creepfill2d_f (size_t nx, size_t ny, float *field, unsigned short repeat, char setWeight, size_t *nChanged)

  Method to fill undefined values in a 2d field in stable time.

- size_t mifi_bad2nanf (float *posPtr, float *endPtr, float badVal)
- size_t mifi_nanf2bad (float *posPtr, float *endPtr, float badVal)
- int mifi_isnanf (float val)
- int mifi_isnand (double val)

16.68.1 Function Documentation

16.68.1.1 int mifi_3d_array_position (int *x*, int *y*, int *z*, int *ix*, int *iy*, int *iz*)

gives the position of an fortran like array of size ix, iy, iz

Returns

the position of x, y, z

16.68.1.2 size_t mifi_bad2nanf (float * *posPtr*, float * *endPtr*, float *badVal*)

Convert bad-values to nan. The mifi_ functions don't handle bad values generally, but forward this work to the floating-point IEEE NaN's. This function converts a general bad value to a nan in a float array.

Parameters

posPtr start pointer of the float array

endPtr end-pointer of the float array (excluded from conversion)

badVal bad value to be converted to nan

Returns

number of conversions

16.68.1.3 int mifi_creepfill2d_f (size_t *nx*, size_t *ny*, float * *field*, unsigned short *repeat*, char *setWeight*, size_t * *nChanged*)

Method to fill undefined values in a 2d field in stable time.

This method will fill undefined values by interpolation of neighboring defined values + the average. A value is assumed to be defined if it is defined in the input field, or if it has been defined through the interpolation method (the defined fields will 'creep' into the undefined area).

The results are very similar to mifi_fill2d_f, but the time will vary only with the size of the undefined area, not with the smoothness of the defined values.

Parameters

nx size of field in x-direction

ny size of field in x-direction

field the data-field to be filled (input/output)

repeat number of times values should be re-smoothed (depending on grid-size, 20-100 (linear with time used)).

setWeight default weight of original values (versus derived values with weight = 1). Must be >= 1, e.g. 2 the higher the value, the smoother the approxamation from the undefined border to average.

nChanged number of changed values (output)

Returns

error-code, usually MIFI_OK

Referenced by MetNoFimex::InterpolatorCreepFill2d::operator()().

16.68.1.4 int mifi_fill2d_f (size_t *nx*, size_t *ny*, float * *field*, float *relaxCrit*, float *corrEff*, size_t *maxLoop*, size_t * *nChanged*)

Method to fill undefined values in a 2d field.

Solves Laplace's equation with Neumann boundary conditions (dA/dn = 0) in rectangular coordinates by an iterative method to fill-in reasonable values at gridpoints containing values with MIFI_UNDEFINED_F or NaNs

Translated to C from Fortran code by H.Engedahl and A.Foss (1990-93).

Parameters

nx size of field in x-direction

ny size of field in x-direction

field the data-field to be filled (input/output)

relaxCrit relaxation criteria. Usually 4 orders of magnitude lower than data in field.

corrEff Coef. of overrelaxation, between +1.2 and +2.0

maxLoop Max. allowed no. of scans in relaxation procedure.

nChanged number of changed values (output)

Returns

error-code, usually MIFI_OK

Referenced by MetNoFimex::InterpolatorFill2d::operator()().

16.68.1.5 int mifi_get_values_bicubic_f (const float * *infield*, float * *outvalues*, const double *x*, const double *y*, const int *ix*, const int *iy*, const int *iz*)

not implemented yet

The bicubic convolution algorithm assigns a value f(x,y) = X ∗ M ∗ F ∗ Mt ∗ Yt with x, y between (0 <= x < 1), X = (1,x, x2, x3), Y = (1, y, y^2, y^3) and F a 4∗4 matrix consisting of the original values of f(-1,-1) to f(2,2).

M is the convolution matrix with a = -0.5 as described by wikipedia (or Catmull-Rom for a = 1, not used here)

Mt and Yt are the transposed matrices/vector.

See also

http://en.wikipedia.org/wiki/Bicubic_interpolation
http://java.sun.com/products/java-media/jai/forDevelopers/jai-apidocs/javax/media/

16.68.1.6 int mifi_get_values_bilinear_f (const float ∗ *infield*, float ∗ *outvalues*, const double *x*, const double *y*, const int *ix*, const int *iy*, const int *iz*)

Bilinear interpolation requires a neighborhood extending one pixel to the right and below the central sample. If the fractional subsample position is given by (xfrac, yfrac), the resampled pixel value will be:

```
(1 - yfrac) * [(1 - xfrac)*s00 + xfrac*s01] +
yfrac       * [(1 - xfrac)*s10 + xfrac*s11]
```

This is documented by the following diagram:

```
s00    s01

    .      < yfrac

s10    s11
    ^
   xfrac
```

See also

http://java.sun.com/products/java-media/jai/forDevelopers/jai-apidocs/javax/media/

Warning

if any of the 4 used values of infield is undefined or outside of infield, the return value will be undefined

16.68.1.7 int mifi_get_values_f (const float ∗ *infield*, float ∗ *outfield*, const double *x*, const double *y*, const int *ix*, const int *iy*, const int *iz*)

Get the nearest neighbor of a value. Values are rounded to array-position.

Parameters

infield 3d fortran array of size ix,iy,iz

outfield 1d array of size iz containing the values

16.68.1.8 int mifi_get_values_linear_d (const double * *infieldA*, const double * *infieldB*, double * *outfield*, const size_t *n*, const double *a*, const double *b*, const double *x*)

This is the same as mifi_get_values_linear_f() for double input/output values.

16.68.1.9 int mifi_get_values_linear_f (const float * *infieldA*, const float * *infieldB*, float * *outfield*, const size_t *n*, const double *a*, const double *b*, const double *x*)

Linear interpolation/extrapolation of values in the arrays infieldA and infieldB at position a and b to a field at outfield at position x with o(x) = in(a) + (x - a) * (in(b) - in(a)) / (b - a) (that describes a linear function o(x) = m*x + c)

This interpolation can be used for linear time-interpolation.

Parameters

infieldA array of size n with values of input at position a

infieldB array of size n with values of input at position b

outfield array of size n with values of input at position x, output

n size of arrays

a position of infieldA

b position of infieldB

x position of outfield

Returns

MIFI_OK return-value set for compatibility with mifi_get_values_log_f()

16.68.1.10 int mifi_get_values_log_f (const float * *infieldA*, const float * *infieldB*, float * *outfield*, const size_t *n*, const double *a*, const double *b*, const double *x*)

Logarithmic interpolation/extrapolation of values in the arrays infieldA and infieldB at position a and b to a field at outfield at position x with o(x) = m*log(x) + c

This interpolation can be used for i.e. log(p)-interpolation. It is tested against results from ncl int2p and vintp2p_ecmwf log(p) interpolation.

Parameters

infieldA array of size n with values of input at position a

infieldB array of size n with values of input at position b

outfield array of size n with values of input at position x, output

n size of arrays

a position of infieldA

b position of infieldB

x position of outfield

Returns

MIFI_OK on success, MIFI_ERROR if log of a, b or x undefined

16.68.1.11 int mifi_get_values_log_log_f (const float * *infieldA*, const float * *infieldB*, float * *outfield*, const size_t *n*, const double *a*, const double *b*, const double *x*)

Log-log interpolation/extrapolation of values in the arrays infieldA and infieldB at position a and b to a field at outfield at position x that describes a function: o(x) = m∗log(log(x)+c

This interpolation can be used for i.e. log(log(p))-interpolation.

Warning

It is tested against results from ncl vintp2p_ecmwf log(log(p)) interpolation, but results vary slightly (∼1%) for unknown reason.

Parameters

infieldA array of size n with values of input at position a

infieldB array of size n with values of input at position b

outfield array of size n with values of input at position x, output

n size of arrays

a position of infieldA

b position of infieldB

x position of outfield

Returns

MIFI_OK on success, MIFI_ERROR if log of a, b or x undefined

16.68.1.12 int mifi_get_vector_reproject_matrix (const char * *proj_input*, const char * *proj_output*, const double * *out_x_axis*, const double * *out_y_axis*, int *out_x_axis_type*, int *out_y_axis_type*, int *ox*, int *oy*, double * *matrix*)

calculate the vector reprojection matrix used in mifi_vector_reproject_values_f

Parameters

method (one of MIFI_VECTOR_KEEP_SIZE, MIFI_VECTOR_RESIZE)

proj_input proj4-string of projection of infield

proj_output proj4-string of projection of outfield

out_x_axis field of size ox. Axis needs to be strong monotonous and if longitude/latitude in degree

out_y_axis field of size oy. Axis needs to be strong monotonous and if longitude/latitude in degree

out_x_axis_type one of MIFI_LATITUDE, MIFI_LONGITUDE, MIFI_PROJ_AXIS

out_y_axis_type one of MIFI_LATITUDE, MIFI_LONGITUDE, MIFI_PROJ_AXIS

ox x-dimension of outfield

oy y-dimension of outfield

matrix matrix of size (4∗ox∗oy)

Returns

MIFI_OK or error value

16.68.1.13 int mifi_interpolate_d (int *method*, char * *proj_input*, double * *infield*, double * *in_x_axis*, double * *in_y_axis*, int *in_x_axis_type*, int *in_y_axis_type*, int *ix*, int *iy*, int *iz*, char * *proj_output*, double * *outfield*, double * *out_x_axis*, double * *out_y_axis*, int *out_x_axis_type*, int *out_y_axis_type*, int *ox*, int *oy*)

not implemented yet

double version of mifi_interpolate_f

See also

mifi_interpolate_f

16.68.1.14 int mifi_interpolate_f (int *method*, const char * *proj_input*, const float * *infield*, const double * *in_x_axis*, const double * *in_y_axis*, const int *in_x_axis_type*, const int *in_y_axis_type*, const int *ix*, const int *iy*, const int *iz*, const char * *proj_output*, float * *outfield*, const double * *out_x_axis*, const double * *out_y_axis*, const int *out_x_axis_type*, const int *out_y_axis_type*, const int *ox*, const int *oy*)

Interpolation between two projections. Missing values are set to MIFI_UNDEFINED_F which is implemented as C99 nanf. The coordinates of a cell give the midpoint of a cell, i.e. cell (10,20) spans ([9.5..10.5[,[19.5-20.5[)

Parameters

method one of MIFI_INTERPOL_NEAREST_NEIGHBOR MIFI_INTERPOL_BILINEAR MIFI_INTERPOL_BICUBIC

proj_input proj4-string of projection of infield

infield real rectangular array of dimension infield[iz,iy,ix]

in_x_axis field of size ix. Axis needs to be strong monotonous and if longitude/latitude in degree

in_y_axis field of size iy. Axis needs to be strong monotonous and if longitude/latitude in degree

in_x_axis_type one of MIFI_LATITUDE, MIFI_LONGITUDE, MIFI_PROJ_AXIS

in_y_axis_type one of MIFI_LATITUDE, MIFI_LONGITUDE, MIFI_PROJ_AXIS

ix x-dimension of infield

iy y-dimension of infield

iz z-dimension of infield and outfield. The z-dim allows you to convert several fields at once without calculating the projection again and again.

proj_output proj4-string of projection of outfield

outfield real rectangular array of dimension outfield[iz,oy,ox]

out_x_axis field of size ox. Axis needs to be strong monotonous and if longitude/latitude in degree

out_y_axis field of size oy. Axis needs to be strong monotonous and if longitude/latitude in degree

out_x_axis_type one of MIFI_LATITUDE, MIFI_LONGITUDE, MIFI_PROJ_AXIS

out_y_axis_type one of MIFI_LATITUDE, MIFI_LONGITUDE, MIFI_PROJ_AXIS

ox x-dimension of outfield

oy y-dimension of outfield

16.68.1.15 int mifi_isnand (double *val*)

check if the value is a nan

Parameters

the value to test

Returns

0 on false, otherwise true

Warning

this function should only be used in C++, which doesn't define the isnan macro defined in C99

16.68.1.16 int mifi_isnanf (float *val*)

check if the value is a nan

Parameters

the value to test

Returns

0 on false, otherwise true

Warning

this function should only be used in C++, which doesn't define the isnan macro defined in C99

16.68.1.17 size_t mifi_nanf2bad (float * *posPtr*, float * *endPtr*, float *badVal*)

Convert nan back to bad-values. See mifi_bad2nanf

Parameters

posPtr start pointer of the float array

endPtr end-pointer of the float array (excluded from conversion)

badVal value NaNs will be converted to

Returns

number of conversions

16.68.1.18 int mifi_points2position (double * *points*, const int *n*, const double * *axis*, const int *num*, const int *axis_type*)

find position in array of position in projection

points2position uses linear splines to find the array-position of points in the given axis

Parameters

points the values will get changed from points in axis coordinates to array coordinates

n number of values in points

axis coordinate axis

num number of elements in coordinate axis

axis_type type of axis, one of MIFI_LONGITUDE, MIFI_LATITUDE, MIFI_PROJ_AXIS

16.68.1.19 int mifi_project_axes (const char * *proj_input*, const char * *proj_output*, const double * *in_x_axis*, const double * *in_y_axis*, const int *ix*, const int *iy*, double * *out_xproj_axis*, double * *out_yproj_axis*)

project axes so that the projetion (x,y) => (x_proj), (y_proj) can be expressed as x_proj(x,y), y_proj(x,y)

all axes must be given or will be returned in radians when converted from/to latlon

Parameters

proj_input input projection proj string

proj_output output projection proj string

in_x_axis x-axis in input-projection

in_y_axis y-axis in input-projection

ix size of x-axis

iy size of y-axis

out_xproj_axis output-values of x_proj(x,y), field needs to be allocated in at least ix*iy size

out_yproj_axis output-values of y_proj(x,y), field needs to be allocated in at least ix*iy size

Returns

error-code

16.68.1.20 int mifi_project_values (const char * *proj_input*, const char * *proj_output*, double * *in_out_x_vals*, double * *in_out_y_vals*, const int *num*)

project values so that the projetion (x,y) => (x_proj), (y_proj) can be expressed as x_proj(x,y), y_proj(x,y)

all values must be given or will be returned in radians when converted from/to latlon

Parameters

proj_input input projection proj string

proj_output output projection proj string

in_out_x_vals x-values, will be input and output

in_out_y_vals y-values, will be input and output

num size of arrays

Returns

error-code

16.68.1.21 int mifi_vector_reproject_values_by_matrix_f (int *method*, const double * *matrix*, float * *u_out*, float * *v_out*, int *ox*, int *oy*, int *oz*)

calculate the reprojected vectors with a known matrix for mifi_vector_reproject_values_f

Parameters

method (one of MIFI_VECTOR_KEEP_SIZE, MIFI_VECTOR_RESIZE)

matrix reprojection matrix of size (4,ox,oy)

u_out values of u, with position in the output-projection (i.e. by prevously applying mifi_interpolate_-f). The values here will be changed!

v_out values of v, with position in the output-projection (i.e. by prevously applying mifi_interpolate_-f). The values here will be changed!

ox x-dimension of outfield

oy y-dimension of outfield

oz z-dimension of the outfield

Returns

MIFI_OK or error value

16.68.1.22 int mifi_vector_reproject_values_f (int *method*, const char * *proj_input*, const char * *proj_output*, float * *u_out*, float * *v_out*, const double * *out_x_axis*, const double * *out_y_axis*, int *out_x_axis_type*, int *out_y_axis_type*, int *ox*, int *oy*, int *oz*)

interpolate the vector values

When reprojecting a vector (i.e. wind (u, v)) from one projection to another, not only the base-position of the vector will change, but also the angle of the vector might change due to rotation and streching within the projection. Thus, the values of (u,v) have to be changed accordingly to projection.

This function allows to only rotate the vector values (MIFI_VECTOR_KEEP_SIZE) which is useful to keep the windspeed constant, even if the projected plane has a different scale, or to completely reproject the vector (MIFI_VECTOR_RESIZE).

This function is implemented by using a first order tailor expansion of the projection: (u', v') = A (u,v) with A a matrix defined at each point (x,y) through

```
proj(x,y)_x' = a11*x+a21*y
proj(x,y)_y' = a12*x+a22*y
```

and the same formulars for (x+delta, y) and (x, y+delta) (with delta a small value against the x or y)

Parameters

method (one of MIFI_VECTOR_KEEP_SIZE, MIFI_VECTOR_RESIZE)

proj_input proj4-string of projection of infield

proj_output proj4-string of projection of outfield

u_out values of u, with position in the output-projection (i.e. by prevously applying mifi_interpolate_-f). The values here will be changed!

v_out values of v, with position in the output-projection (i.e. by prevously applying mifi_interpolate_-f). The values here will be changed!

out_x_axis field of size ox. Axis needs to be strong monotonous and if longitude/latitude in degree

out_y_axis field of size oy. Axis needs to be strong monotonous and if longitude/latitude in degree

out_x_axis_type one of MIFI_LATITUDE, MIFI_LONGITUDE, MIFI_PROJ_AXIS

out_y_axis_type one of MIFI_LATITUDE, MIFI_LONGITUDE, MIFI_PROJ_AXIS

ox x-dimension of outfield

oy y-dimension of outfield

oz z-dimension of the outfield

Returns

MIFI_OK or error value

16.69 include/fimex/Logger.h File Reference

```
#include <boost/shared_ptr.hpp>
#include <string>
#include <sstream>
```

Classes

- class MetNoFimex::Logger

Namespaces

- namespace MetNoFimex

Defines

- #define LOG4FIMEX(logger, level, message)

Typedefs

- typedef boost::shared_ptr< Logger > MetNoFimex::LoggerPtr

Functions

- Logger::LogLevel MetNoFimex::defaultLogLevel ()
- void MetNoFimex::defaultLogLevel (Logger::LogLevel)
- LoggerPtr MetNoFimex::getLogger (const **std::string** &className)

16.69.1 Define Documentation

16.69.1.1 #define LOG4FIMEX(logger, level, message)

Value:

```
{ \
    if (logger->isEnabledFor(level)) {\
        std::ostringstream buffer; \
        buffer << message; \
        logger->forcedLog(level, buffer.str(), __FILE__, __LINE__);}}
```

use this pragma to log a message of a level

Parameters

logger a logger as retrieved with getLogger("com.bar")

level a fimex LogLevel, i.e. OFF, FATAL, ERROR, WARN, INFO, DEBUG

message the message to log

16.70 include/fimex/MetGmCDMReader.h File Reference

```
#include "fimex/CDMReader.h"
#include "fimex/XMLInput.h"
#include <boost/shared_ptr.hpp>
```

Classes

- class MetNoFimex::MetGmCDMReader

Namespaces

- namespace MetNoFimex

16.71 include/fimex/MetGmCDMWriter.h File Reference

```
#include "fimex/CDMWriter.h"
```

Classes

- class MetNoFimex::MetGmCDMWriter

Namespaces

- namespace MetNoFimex

16.72 include/fimex/mifi_cdm_reader.h File Reference

```
#include <boost/shared_ptr.hpp>
#include "fimex/CDMReader.h"
```

Classes

- class mifi_cdm_reader

16.73 include/fimex/mifi_constants.h File Reference

```
#include <math.h>
#include <stddef.h>
```

Defines

- #define MIFI_PI 3.14159265358979323846264338327 95
- #define MIFI_INTERPOL_NEAREST_NEIGHBOR 0

  interpolation method

- #define MIFI_INTERPOL_BILINEAR 1

  interpolation method

- #define MIFI_INTERPOL_BICUBIC 2

  interpolation method

- #define MIFI_INTERPOL_COORD_NN 3

  interpolation method

- #define MIFI_INTERPOL_COORD_NN_KD 4

  interpolation method

- #define MIFI_INTERPOL_FORWARD_SUM 5

  interpolation method

- #define MIFI_INTERPOL_FORWARD_MEAN 6

  interpolation method

- #define MIFI_INTERPOL_FORWARD_MEDIAN 7

  interpolation method

- #define MIFI_INTERPOL_FORWARD_MAX 8

  interpolation method

- #define MIFI_INTERPOL_FORWARD_MIN 9

  interpolation method

- #define MIFI_VECTOR_KEEP_SIZE 0

  vector projection flag

- #define MIFI_VECTOR_RESIZE 1

  vector projection flag

- #define MIFI_VINT_PRESSURE 0

  vertical interpolation type

- #define MIFI_VINT_HEIGHT 1

  vertical interpolation type

- #define MIFI_VINT_METHOD_LIN 0
  vertical interpolation method
- #define MIFI_VINT_METHOD_LOG 1
  vertical interpolation method
- #define MIFI_VINT_METHOD_LOGLOG 2
  vertical interpolation method
- #define MIFI_UNDEFINED_F (nanf(""))
  undefined value for floats
- #define MIFI_UNDEFINED_D (nan(""))
  undefined value for doubles
- #define MIFI_ERROR -1
  return code, error
- #define MIFI_OK 1
  return code, ok
- #define MIFI_PROJ_AXIS 0
  projection axis in m-equivalent
- #define MIFI_LONGITUDE 1
  longitude projection axis in degrees
- #define MIFI_LATITUDE 2
  latitude projection axis in degrees
- #define MIFI_DEBUG 0
  debug flag

16.73.1 Define Documentation

16.73.1.1 #define MIFI_DEBUG 0

debug flag

16.73.1.2 #define MIFI_ERROR -1

return code, error

16.73.1.3 #define MIFI_INTERPOL_BICUBIC 2

interpolation method

Flag for bicubic interpolation. This requires, that the original data comes with a properly defined projection, i.e. implicit as latlon or explicit with projection-string

16.73.1.4 #define MIFI_INTERPOL_BILINEAR 1

interpolation method

Flag for bilinear interpolation. This requires, that the original data comes with a properly defined projection, i.e. implicit as latlon or explicit with projection-string

16.73.1.5 #define MIFI_INTERPOL_COORD_NN 3

interpolation method

Flag for nearest neighbor interpolation using lon/lat coordinates rather than the input projection. This is largely a brute force method which may take long time.

Vector projection is not implemented (not defined?)

Warning

this works only from CDMInterpolator

16.73.1.6 #define MIFI_INTERPOL_COORD_NN_KD 4

interpolation method

Flag for nearest neighbor interpolation using coordinates with KD-tree. This works as nearest neighbor in the output-projection and has therefore numerical problems in some points, i.e. near southpole when using northpole-polarstereographic.

It doesn't work with output projections in degree, i.e. rotated latitude longitude, since distances are calculated as outX^2∗outY^2

Vector projection is not implemented (not defined?)

Warning

this works only from CDMInterpolator

16.73.1.7 #define MIFI_INTERPOL_FORWARD_MAX 8

interpolation method

forward interpolation, maximum over all matching defined input-cells

16.73.1.8 #define MIFI_INTERPOL_FORWARD_MEAN 6

interpolation method

forward interpolation, averaging (mean) over all matching defined input-cells

16.73.1.9 #define MIFI_INTERPOL_FORWARD_MEDIAN 7

interpolation method

forward interpolation, median over all matching defined input-cells

16.73.1.10 #define MIFI_INTERPOL_FORWARD_MIN 9

interpolation method

forward interpolation, minimum over all matching defined input-cells

16.73.1.11 #define MIFI_INTERPOL_FORWARD_SUM 5

interpolation method

forward interpolation, summing over all matching input-cells

16.73.1.12 #define MIFI_INTERPOL_NEAREST_NEIGHBOR 0

interpolation method

Flag for nearest neighbor interpolation. This requires, that the original data comes with a properly defined projection, i.e. implicit as latlon or explicit with projection-string

16.73.1.13 #define MIFI_LATITUDE 2

latitude projection axis in degrees

16.73.1.14 #define MIFI_LONGITUDE 1

longitude projection axis in degrees

16.73.1.15 #define MIFI_OK 1

return code, ok

16.73.1.16 #define MIFI_PI 3.14159265358979323846264338327 95

M_PI is no longer part of C99, so it needs to be declared for fimex

16.73.1.17 #define MIFI_PROJ_AXIS 0

projection axis in m-equivalent

16.73.1.18 #define MIFI_UNDEFINED_D (nan(""))

undefined value for doubles

16.73.1.19 #define MIFI_UNDEFINED_F (nanf(""))

undefined value for floats

16.73.1.20 #define MIFI_VECTOR_KEEP_SIZE 0

vector projection flag

new size will be like old size

16.73.1.21 #define MIFI_VECTOR_RESIZE 1

vector projection flag

vector might change size with projection

16.73.1.22 #define MIFI_VINT_HEIGHT 1

vertical interpolation type

vertical interpolation to height above ground levels in m

16.73.1.23 #define MIFI_VINT_METHOD_LIN 0

vertical interpolation method

linear interpolation, e.g. mifi_get_values_lin_f()

16.73.1.24 #define MIFI_VINT_METHOD_LOG 1

vertical interpolation method

logarithmic interpolation, e.g. mifi_get_values_log_f()

16.73.1.25 #define MIFI_VINT_METHOD_LOGLOG 2

vertical interpolation method

double logarithmic interpolation, e.g. mifi_get_values_log_f()

16.73.1.26 #define MIFI_VINT_PRESSURE 0

vertical interpolation type

vertical interpolation to pressure levels in hPa

16.74 include/fimex/NcmlCDMReader.h File Reference

```
#include "fimex/CDMReader.h"
#include "fimex/CDMDataType.h"
#include "fimex/XMLInput.h"
#include <map>
```

Classes

- class MetNoFimex::NcmlCDMReader

Namespaces

- namespace MetNoFimex

16.75 include/fimex/NetCDF_CDMReader.h File Reference

```
#include "fimex/CDMReader.h"
```

Classes

- class MetNoFimex::NetCDF_CDMReader

Namespaces

- namespace MetNoFimex

16.76 include/fimex/NetCDF_CDMWriter.h File Reference

```
#include "fimex/CDMWriter.h"
#include "fimex/CDM.h"
#include <map>
#include <string>
```

Classes

- class MetNoFimex::NetCDF_CDMWriter

Namespaces

- namespace MetNoFimex

16.77 include/fimex/Null_CDMWriter.h File Reference

```
#include "fimex/CDMWriter.h"
```

Classes

- class MetNoFimex::Null_CDMWriter

Namespaces

- namespace MetNoFimex

16.78 include/fimex/ReplaceStringObject.h File Reference

```
#include <iostream>
#include <string>
#include <vector>
```

Classes

- class MetNoFimex::ReplaceStringObject

Namespaces

- namespace MetNoFimex

16.79 include/fimex/ReplaceStringTimeObject.h File Reference

```
#include "fimex/ReplaceStringObject.h"
#include <ctime>
```

Classes

- class MetNoFimex::ReplaceStringTimeObject

Namespaces

- namespace MetNoFimex

16.80 include/fimex/SliceBuilder.h File Reference

```
#include <string>
#include <vector>
#include <map>
#include <set>
#include <boost/shared_ptr.hpp>
```

Classes

- class MetNoFimex::SliceBuilder

Namespaces

- namespace MetNoFimex

16.81 include/fimex/SpatialAxisSpec.h File Reference

```
#include "fimex/TimeUnit.h"
#include "fimex/CDMException.h"
#include <vector>
#include <string>
```

Classes

- class MetNoFimex::SpatialAxisSpec

Namespaces

- namespace MetNoFimex

16.82 include/fimex/TimeLevelDataSliceFetcher.h File Reference

```
#include <boost/shared_ptr.hpp>
#include <vector>
#include "fimex/Logger.h"
#include "fimex/CDMException.h"
```

Classes

- class MetNoFimex::TimeLevelDataSliceFetcher

  read a slice of a given time/level combination from a cdmReader

Namespaces

- namespace MetNoFimex

16.83 include/fimex/TimeSpec.h File Reference

```
#include "fimex/TimeUnit.h"
#include "fimex/CDMException.h"
#include <vector>
#include <string>
```

Classes

- class MetNoFimex::TimeSpec

Namespaces

- namespace MetNoFimex

16.84 include/fimex/TimeUnit.h File Reference

```
#include "boost/shared_ptr.hpp"
#include "fimex/Units.h"
#include "fimex/CDMException.h"
#include <iostream>
#include <boost/date_time/posix_time/posix_time_types.hpp>
```

Classes

- class MetNoFimex::FimexTime
- class MetNoFimex::TimeUnit

Namespaces

- namespace MetNoFimex

Functions

- **std::ostream** & MetNoFimex::operator<< (**std::ostream** &out, const FimexTime &fTime)

  minimum FimexTime

- FimexTime MetNoFimex::string2FimexTime (const **std::string** &str) throw (CDMException)

16.85 include/fimex/Units.h File Reference

```
#include <string>
#include "fimex/CDMException.h"
```

Classes

- class MetNoFimex::UnitException
- class MetNoFimex::Units

Namespaces

- namespace MetNoFimex

Functions

- void MetNoFimex::handleUdUnitError (int unitErrCode, const **std::string** &message="") throw (UnitException)

16.86 include/fimex/Utils.h File Reference

```
#include <vector>
#include <utility>
#include <iterator>
#include <sstream>
#include <cmath>
#include <boost/date_time/posix_time/posix_time_types.hpp>
#include <limits>
#include "fimex/CDMException.h"
#include <boost/shared_array.hpp>
```

Classes

- struct MetNoFimex::staticCast< OUT >
- class MetNoFimex::ScaleValue< IN, OUT >
- class MetNoFimex::ChangeMissingValue< IN, OUT >
- struct MetNoFimex::SharedArrayConstCastDeleter< T >

Namespaces

- namespace MetNoFimex

Typedefs

- typedef long MetNoFimex::epoch_seconds

Functions

- int MetNoFimex::round (double num)
- **std::string** MetNoFimex::trim (const **std::string** &str)
- template<class InputIterator >
  std::string MetNoFimex::join (InputIterator start, InputIterator end, **std::string** delim=",")
- template<typename InputIterator >
  std::pair< typename **std::iterator_traits**< InputIterator >::difference_type, typename **std::iterator_traits**< InputIterator >::difference_type > MetNoFimex::find_closest_distinct_elements (InputIterator start, InputIterator end, double x)
- template<typename InputIterator >
  std::pair< typename **std::iterator_traits**< InputIterator >::difference_type, typename **std::iterator_traits**< InputIterator >::difference_type > MetNoFimex::find_closest_neighbor_distinct_elements (InputIterator start, InputIterator end, double x)
- template<class InputIterator >
  std::string MetNoFimex::joinPtr (InputIterator start, InputIterator end, **std::string** delim=",")
- **std::vector**< **std::string** > MetNoFimex::tokenize (const **std::string** &str, const **std::string** &delimiters=" ")
- **std::string** MetNoFimex::string2lowerCase (const **std::string** &str)

- template<typename T >
  std::string MetNoFimex::type2string (T in)
- template<>
  std::string MetNoFimex::type2string< double > (double in)
- template<typename T >
  T MetNoFimex::string2type (**std::string** s)
- epoch_seconds MetNoFimex::posixTime2epochTime (const boost::posix_time::ptime &time)
- template<typename T >
  std::vector< T > MetNoFimex::tokenizeDotted (const **std::string** &str, const **std::string** &delimiter=",") throw (CDMException)
- template<typename T >
  boost::shared_array< const T > MetNoFimex::makeSharedArrayConst (const boost::shared_array< T > &sa)

16.87 include/fimex/vertical_coordinate_transformations.h File Reference

```
#include "fimex/mifi_constants.h"
```

Functions

- int mifi_atmosphere_ln_pressure (size_t n, double p0, const double *lev, double *pressure)
- int mifi_atmosphere_sigma_pressure (size_t n, double ptop, double ps, const double *lev, double *pressure)
- int mifi_atmosphere_hybrid_sigma_pressure (size_t n, double p0, double ps, const double *a, const double *b, double *pressure)
- int mifi_atmosphere_hybrid_sigma_ap_pressure (size_t n, double ps, const double *ap, const double *b, double *pressure)
- int mifi_barometric_pressure (size_t n, double P_b, const double *h, double T_b, double *pressure)
- int mifi_barometric_standard_pressure (size_t n, const double *h, double *pressure)
- int mifi_barometric_height (size_t n, double P_b, const double *p, double T_b, double *height)
- int mifi_barometric_standard_height (size_t n, const double *p, double *height)
- int mifi_omega_to_vertical_wind (size_t n, const double *omega, const double *p, const double *t, double *w)

16.87.1 Function Documentation

16.87.1.1 int mifi_atmosphere_hybrid_sigma_ap_pressure (size_t *n*, double *ps*, const double * *ap*, const double * *b*, double * *pressure*)

convert a standard_name="atmosphere_hybrid_sigma_pressure_coordinate" to pressure using the formular p(k) = ap(k) + b(k)*ps

This is the same as mifi_atmosphere_hybrid_sigma_pressure(), but with the reference pressure and a joined already. Choice depends on the model, i.e. available input values.

Parameters

n size of arrays ap, b and pressure

ps surface pressure - usually varying in time,x,y

ap pressure level values

b dimensionless level values

pressure output values in the same unit as p0 and ps and at the same place as ps

Returns

MIFI_OK on success or MIFI_ERROR on failure

16.87.1.2 int mifi_atmosphere_hybrid_sigma_pressure (size_t *n*, double *p0*, double *ps*, const double * *a*, const double * *b*, double * *pressure*)

convert a standard_name="atmosphere_hybrid_sigma_pressure_coordinate" to pressure using the formular p(k) = a(k)*p0 + b(k)*ps

Parameters

n size of arrays a, b and pressure

p0 reference pressure

ps surface pressure - usually varying in time,x,y

a dimensionless level values

b dimensionless level values

pressure output values in the same unit as p0 and ps and at the same place as ps

Returns

MIFI_OK on success or MIFI_ERROR on failure

16.87.1.3 int mifi_atmosphere_ln_pressure (size_t *n*, double *p0*, const double * *lev*, double * *pressure*)

convert a standard_name="atmosphere_ln_pressure_coordinate" to pressure using the formular p(k) = p0 * exp(-lev(k))

Parameters

n size of arrays lev and pressure

p0 base pressure

lev level values

pressure output values in the same unit as p0

Returns

MIFI_OK on success or MIFI_ERROR on failure

16.87.1.4 int mifi_atmosphere_sigma_pressure (size_t *n*, double *ptop*, double *ps*, const double * *lev*, double * *pressure*)

convert a standard_name="atmosphere_sigma_coordinate" to pressure using the formular p(k) = ptop + sigma(k)*(ps-ptop)

Parameters

n size of arrays sigma and pressure

ptop pressure on model top layer (constant for a model)

ps surface pressure - usually varying in time,x,y

sigma level values

pressure output values in the same unit as ptop and ps and at the same place as ps

Returns

MIFI_OK on success or MIFI_ERROR on failure

16.87.1.5 int mifi_barometric_height (size_t *n*, double *P_b*, const double * *p*, double *T_b*, double * *height*)

convert pressure to height using the inverse formular http://en.wikipedia.org/wiki/Barometric_formula

h(k) = -R*T_b/g*M * log(p(k)/P_b);

with P_b and T_b pressure and temperature at the layer b (i.e. surface)

g = 9.80665 m/s2 M = Molar mass of Earth's air (0.0289644 kg/mol) R = Universal gas constant (8.31432 N·m /(mol·K))

Parameters

n size of array h and pressure

P_b pressure at base-layer (i.e. surface, or means-sea-level) - usually varying in time,x,y

p pressure at level

T_b temperature at base layer in K - usually varying in time,x,y

height output values, height above base_layer in m

Warning

This function has not been tested against possibly existing implementations

16.87.1.6 int mifi_barometric_pressure (size_t *n*, double *P_b*, const double * *h*, double *T_b*, double * *pressure*)

convert height to pressure using the formular http://en.wikipedia.org/wiki/Barometric_formula

P(h) = P_b exp[-gM/R * h/T_b]

with P_b and T_b pressure and temperature at the layer b (i.e. surface) and h_b the height above the layer b

g = 9.80665 m/s2 M = Molar mass of Earth's air (0.0289644 kg/mol) R = Universal gas constant (8.31432 N·m /(mol·K))

Parameters

n size of array h and pressure

P_b pressure at base-layer (i.e. surface, or means-sea-level) - usually varying in time,x,y

h height in m above base-layer

T_b temperature at base layer in K - usually varying in time,x,y

pressure output values in the same unit as p_b and at the same place as ps

Warning

This function has not been tested against possibly existing implementations

16.87.1.7 int mifi_barometric_standard_height (size_t *n*, const double * *p*, double * *height*)

convert pressure to height using the formular http://en.wikipedia.org/wiki/Barometric_formula and using the international standard atmosphere http://en.wikimedia.org/wiki/International_Standard_Atmosphere

16.87.1.8 int mifi_barometric_standard_pressure (size_t *n*, const double * *h*, double * *pressure*)

convert height to pressure using the formular http://en.wikipedia.org/wiki/Barometric_formula and using the international standard atmosphere http://en.wikimedia.org/wiki/International_Standard_Atmosphere

16.87.1.9 int mifi_omega_to_vertical_wind (size_t *n*, const double * *omega*, const double * *p*, const double * *t*, double * *w*)

convert the vertical pressure change omega to vertical wind-speed using omega = - rho * g * w

Parameters

n size of the array omega, p, t, w

omega vertical flow in Pa/s

p pressure in Pa

t temperature in K

w output-array for vertical wind speed in m/s

Returns

MIFI_OK status

16.88 include/fimex/WdbCDMReader.h File Reference

```
#include "fimex/CDMReader.h"
#include "fimex/XMLInput.h"
#include <string>
```

Classes

- class MetNoFimex::WdbCDMReader

Namespaces

- namespace MetNoFimex

16.89 include/fimex/XMLDoc.h File Reference

```
#include <boost/utility.hpp>
#include <boost/shared_ptr.hpp>
#include <string>
#include "fimex/CDMException.h"
```

Classes

- class MetNoFimex::XMLDoc

Namespaces

- namespace MetNoFimex

Typedefs

- typedef struct _xmlDoc xmlDoc
- typedef struct _xmlNode xmlNode
- typedef struct _xmlXPathContext xmlXPathContext
- typedef xmlNode * xmlNodePtr
- typedef struct _xmlXPathObject xmlXPathObject
- typedef boost::shared_ptr< xmlXPathObject > MetNoFimex::XPathObjPtr

Functions

- **std::string** MetNoFimex::getXmlProp (const xmlNodePtr node, const **std::string** &attrName)
- **std::string** MetNoFimex::getXmlName (const xmlNodePtr node)
- **std::string** MetNoFimex::getXmlContent (const xmlNodePtr node)

  get all text-contents of the node or underlying nodes

16.89.1 Typedef Documentation

16.89.1.1 typedef struct _xmlDoc xmlDoc

16.89.1.2 typedef struct _xmlNode xmlNode

16.89.1.3 typedef xmlNode* xmlNodePtr

16.89.1.4 typedef struct _xmlXPathContext xmlXPathContext

16.89.1.5 typedef struct _xmlXPathObject xmlXPathObject

16.90 include/fimex/XMLInput.h File Reference

```
#include <string>
#include <boost/shared_ptr.hpp>
```

Classes

- class MetNoFimex::XMLInput
- class MetNoFimex::XMLInputFile
- class MetNoFimex::XMLInputString
- class MetNoFimex::XMLInputURL

Namespaces

- namespace MetNoFimex

Chapter 17

Example Documentation

17.1 coordinateSystem.cpp

Example on using the CoordinateSystem in combination with a CDMReader.

```
#include "fimex/coordSys/CoordinateSystem.h"
#include "fimex/CoordinateSystemSliceBuilder.h"
#include "fimex/NetCDF_CDMReader.h"

using namespace MetNoFimex;
using namespace std;

int main(int argc, char* args[]) {
    boost::shared_ptr<CDMReader> reader(new NetCDF_CDMReader("coordTest.nc"));
    //boost::shared_ptr<CDMReader> reader(new NetCDF_CDMReader("coordRefTimeTest.
      nc"));
    const CDM& cdm = reader->getCDM();

    // get all coordinate systems from file, usually one, but may be a few (theor
      etical limit: # of variables)
    vector<boost::shared_ptr<const CoordinateSystem> > coordSys =
      listCoordinateSystems(cdm);
    // find an appropriate coordinate system for a variable
    string varName = "air_temperature";
    vector<boost::shared_ptr<const CoordinateSystem> >::iterator varSysIt =
            find_if(coordSys.begin(), coordSys.end(),
      CompleteCoordinateSystemForComparator(varName));
    if (varSysIt != coordSys.end()) {
        if ((*varSysIt)->isSimpleSpatialGridded()) {
            CoordinateSystem::ConstAxisPtr xAxis = (*varSysIt)->getGeoXAxis(); //
       X or Lon
            CoordinateSystem::ConstAxisPtr yAxis = (*varSysIt)->getGeoYAxis(); //
       Y or Lat
            CoordinateSystem::ConstAxisPtr tAxis = (*varSysIt)->getTimeAxis(); //
       time

            CoordinateSystemSliceBuilder sb(cdm, *varSysIt);
            // handling of time
            if (tAxis.get() != 0) {
                // time-Axis, eventually multi-dimensional, i.e. forecast_referen
      ce_time
                if ((*varSysIt)->hasAxisType(CoordinateAxis::ReferenceTime)) {
                    CoordinateSystem::ConstAxisPtr rtAxis = (*varSysIt)->findAxis
      OfType(CoordinateAxis::ReferenceTime);
                    boost::shared_ptr<Data> refTimes = reader->getScaledDataInUni
      t(rtAxis->getName(),"seconds since 1970-01-01 00:00:00");
```

```
                /* do something with the refTimes and select the wanted Posit
    ion */
                size_t refTimePos = 3; /* or whatever you select between 0 (d
    efault) and refTimes->size()-1 */
                sb.setReferenceTimePos(refTimePos);
            }
            boost::shared_ptr<Data> times = reader->getDataSlice(tAxis->getNa
    me(), sb.getTimeVariableSliceBuilder());
            /* select the desired startTime and the sice for the time-slices
    */
            // fetch the 2nd and 3rd time-step of the 4th run
            sb.setTimeStartAndSize(1, 2); // default is all of ReferenceTimeP
    os
        }


        // further selection of data
        // select 3-7 y-points
        sb.setStartAndSize(yAxis, 3, 5);
        sb.setAll(xAxis);

        // by default, all other dimensions are fetched at maximum size
        // here, I reduce them to the first slice
        vector<string> dims = sb.getUnsetDimensionNames();
        for (vector<string>::iterator dim = dims.begin(); dim != dims.end();
    ++dim) {
            sb.setStartAndSize(*dim, 0, 1);
        }

        // fetch the data
        boost::shared_ptr<Data> data = reader->getDataSlice(varName, sb);
        /* do something with the data */
      }
    }
    return 0;
}
```

Index