MI - Fimex

Generated by Doxygen 1.5.5

Tue Jan 13 12:32:08 2009

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
- gribWriter Configuration
- netcdfWriter 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 intput file
  --input.config arg                  non-standard input configuration
  --input.printNcML                   print NcML description of input file
  --output.file arg                   output file
  --output.type arg                   filetype of output file
  --output.config arg                 non-standard output configuration
  --extract.removeVariable arg        remove 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.printNcML                 print NcML description of extractor
  --interpolate.projString arg        proj4 input string describing the new projection
  --interpolate.method arg            interpolation method, one of nearestneighbor,
                                      bilinear or bicubic
  --interpolate.xAxisValues arg       string with values on x-Axis, use ... to
                                      continue, i.e. 10.5,11,...,29.5
  --interpolate.yAxisValues arg       string with values on x-Axis, use ... to
                                      continue, i.e. 10.5,11,...,29.5
  --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.printNcML             print NcML description 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
```

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

```
# config file for program fimex
[input]
file=flth00.dat
config=../share/etc/felt2nc_variables.xml
type=felt

[output]
```

```
file=test.nc
type=netcdf

[extract]
removeVariable=relative_humidity

[extract.reduceDimension]
name=x
start=2
end=-2

[extract.reduceDimension]
name=y
start=2
end=-2

[interpolate]
method = bilinear
projString = +proj=stere +lat_0=90 +lon_0=-32 +lat_ts=60 +elips=sphere +a=6371000 +e=0
xAxisValues = -500000,-450000,...,5000000
yAxisValues = -500000,-450000,...,5000000
xAxisUnit = m
yAxisUnit = m

[timeInterpolate]
timeSpec = 0,3,...,x;relativeUnit=hours since 2001-01-01 10:00:00;unit=hours since 2007-05-16 00:00:00
```

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.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE cdm_felt_config SYSTEM "felt2nc_variables.dtd">
<cdm_felt_config>
<!-- optional processing options for felt-files -->
<processOptions>
    <!-- allowed deviation of gridParameter (6 values, see felt.doc) to still assumed to be the same grid,
<!--  <option name="gridParameterDelta" value="0 0 0 0 0 0"/> -->
</processOptions>
<global_attributes>
<attribute name="Conventions" value="CF-1.0" type="string" />
<attribute name="institution" value="Norwegian Meteorological Institute, met.no" type="string" />
<attribute name="source" value="HIRLAM" type="string" />
<attribute name="title" value="unknown" type="string" />
<attribute name="min_time" value="%MIN_DATETIME(%Y-%m-%d %H:%M:%SZ)%" type="string" />
<attribute name="max_time" value="%MAX_DATETIME(%Y-%m-%d)%" type="string" />
```

```
<attribute name="Expires" value="%MAX_DATETIME(%Y-%m-%d,2419200)%" type="string" />
<attribute name="references" value="unknown" type="string" />
<attribute name="history" value="unknown" type="string" />
<attribute name="comment" value="none" type="string" />
</global_attributes>
<axes>
<time id="time" name="time" type="double">
<attribute name="long_name" value="time" type="string" />
<attribute name="standard_name" value="time" type="string" />
<attribute name="units" value="seconds since 1970-01-01 00:00:00 +00:00" type="string" />
</time>
<!-- polar-stereographic at 60deg -->
<spatial_axis projection_felt_id="1" id="x" name="x" type="int">
<attribute name="long_name" value="x-coordinate in Cartesian system" type="string" />
<attribute name="standard_name" value="projection_x_coordinate" type="string" />
<attribute name="units" value="m" type="string" />
</spatial_axis>
<spatial_axis projection_felt_id="1" id="y" name="y" type="int">
<attribute name="long_name" value="y-coordinate in Cartesian system" type="string" />
<attribute name="standard_name" value="projection_y_coordinate" type="string" />
<attribute name="units" value="m" type="string" />
</spatial_axis>
<!-- geographic -->
<spatial_axis projection_felt_id="2" id="x" name="lon" type="float">
<attribute name="long_name" value="longitude" type="string" />
<attribute name="units" value="degrees_east" type="string" />
</spatial_axis>
<spatial_axis projection_felt_id="2" id="y" name="lat" type="float">
<attribute name="long_name" value="latitude" type="string" />
<attribute name="units" value="degrees_north" type="string" />
</spatial_axis>
<!-- spherical rotated -->
<spatial_axis projection_felt_id="3" id="x" name="rlon" type="float">
<attribute name="long_name" value="rotated longitude" type="string" />
<attribute name="standard_name" value="grid_longitude" type="string" />
<attribute name="units" value="degrees" type="string" />
</spatial_axis>
<spatial_axis projection_felt_id="3" id="y" name="rlat" type="float">
<attribute name="long_name" value="rotated latitude" type="string" />
<attribute name="standard_name" value="grid_latitude" type="string" />
<attribute name="units" value="degrees" type="string" />
</spatial_axis>
<!-- polar-stereographic -->
<spatial_axis projection_felt_id="4" id="x" name="x" type="int">
<attribute name="long_name" value="x-coordinate in Cartesian system" type="string" />
<attribute name="standard_name" value="projection_x_coordinate" type="string" />
<attribute name="units" value="m" type="string" />
</spatial_axis>
<spatial_axis projection_felt_id="4" id="y" name="y" type="int">
<attribute name="long_name" value="y-coordinate in Cartesian system" type="string" />
<attribute name="standard_name" value="projection_y_coordinate" type="string" />
<attribute name="units" value="m" type="string" />
</spatial_axis>
<!-- mercator -->
<spatial_axis projection_felt_id="5" id="x" name="x" type="int">
<attribute name="long_name" value="x-coordinate in Cartesian system" type="string" />
<attribute name="standard_name" value="projection_x_coordinate" type="string" />
<attribute name="units" value="m" type="string" />
</spatial_axis>
<spatial_axis projection_felt_id="5" id="y" name="y">
<attribute name="long_name" value="y-coordinate in Cartesian system" type="string" />
<attribute name="standard_name" value="projection_y_coordinate" type="string" />
<attribute name="units" value="m" type="string" />
</spatial_axis>
<spatial_axis id="longitude" name="longitude">
<attribute name="valid_max" value="180." type="float" />
<attribute name="valid_min" value="-180." type="float" />
```

```
<attribute name="long_name" value="longitude" type="string" />
<attribute name="standard_name" value="longitude" type="string" />
<attribute name="units" value="degree_east" type="string" />
</spatial_axis>
<spatial_axis id="latitude" name="latitude">
<attribute name="valid_max" value="90." type="float" />
<attribute name="valid_min" value="-90." type="float" />
<attribute name="long_name" value="latitude" type="string" />
<attribute name="standard_name" value="latitude" type="string" />
<attribute name="units" value="degree_north" type="string" />
</spatial_axis>
<vertical_axis id="pressure" name="pressure" felt_id="1" type="short">
<attribute name="description" value="pressure" type="string" />
<attribute name="long_name" value="pressure" type="string" />
<attribute name="standard_name" value="air_pressure" type="string" />
<attribute name="positive" value="up" type="string" />
<attribute name="units" value="hPa" type="string" />
</vertical_axis>
<vertical_axis id="sigma" name="sigma" felt_id="2" type="short">
<attribute name="description" value="atmosphere sigma coordinate" type="string" />
<attribute name="long_name" value="atmosphere_sigma_coordinate" type="string" />
<attribute name="standard_name" value="atmosphere_sigma_coordinate" type="string" />
<attribute name="positive" value="up" type="string" />
<attribute name="scale_factor" value="0.001" type="float" />
</vertical_axis>
<vertical_axis id="h" name="h" felt_id="3" type="short">
<attribute name="description" value="vertical coordinate used for wave variables, value: 0" type="string"
<attribute name="long_name" value="vertical_wave_coordinate" type="string" />
</vertical_axis>
<vertical_axis id="theta" name="theta" felt_id="4" type="short">
<attribute name="description" value="isentrop layer?" type="string" />
</vertical_axis>
<vertical_axis id="depth" name="depth" felt_id="5" type="short">
<attribute name="description" value="geopotential level relative to equilibrium surface" type="string" />
<attribute name="long_name" value="depth" type="string" />
<attribute name="positive" value="down" type="string" />
<attribute name="standard_name" value="depth" type="string" />
<attribute name="units" value="m" type="string" />
</vertical_axis>
<vertical_axis id="layer" name="layer" felt_id="6" type="short">
<attribute name="description" value="ocean model layer no., isopycnic or hybrid" type="string" />
<attribute name="long_name" value="ocean_layer_coordinate" type="string" />
<attribute name="positive" value="down" type="string" />
</vertical_axis>
<vertical_axis id="sigma" name="sigma" felt_id="7" type="short">
<attribute name="description" value="ocean sigma coordinate, surface is 0, bottom is 1" type="string" />
<attribute name="long_name" value="ocean_sigma_coordinate" type="string" />
<attribute name="positive" value="down" type="string" />
<attribute name="standard_name" value="ocean_sigma_coordinate" type="string" />
<attribute name="scale_factor" value="0.001" type="float" />
</vertical_axis>
<vertical_axis id="surface" name="surface" felt_id="8" type="short">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="description" value="ocean surface, or vertically integrated" type="string" />
<attribute name="long_name" value="model_surface" type="string" />
<attribute name="positive" value="up" type="string" />
</vertical_axis>
<vertical_axis id="k" name="k" felt_id="10" type="double">
<attribute name="standard_name" value="atmosphere_hybrid_sigma_pressure_coordinate" type="string" />
<attribute name="formular" value="p(n,k,j,i) = ap(k) + b(k)*ps(n,j,i)" type="string" />
<attribute name="formula_terms" value="ap: ap b: b ps: ps p0: p0" type="string" />
<attribute name="long_name" value="atmosphere_hybrid_sigma_pressure_coordinate" type="string" />
<attribute name="positive" value="up" type="string" />
<values mode="hybridSigmaCalc(ap,b)" />
<!--  optional values, will otherwise be calculated -->
<!-- <values mode="inline">0.01000025677 0.030167302165 0.0506574118 0.071450009935 0.09252333188 0.113854
<additional_axis_variable name="ap" type="double" axis="k">
```

```
<attribute name="units" value="Pa" type="string" />
<values mode="level2" scale_factor="10" />
<!--  optional values, will otherwise be retrieved from level2 -->
<!--  <values>1000.025677 3016.7302165 5053.90618 7087.0199935 9093.765188 11053.98013 12949.566675 14764.
</additional_axis_variable>
<additional_axis_variable name="b" type="double" axis="k">
<attribute name="units" value="1" type="string" />
<values mode="hybridLevels" scale_factor="0.0001" />
<!--  optional values, will otherwise be retrieved from ident19 -->
<!--  <values>0 0 0.00011835 0.00057981 0.00158568 0.003314615 0.00592347 0.00954814 0.01430438 0.02028863
</additional_axis_variable>
<!-- currently not possible to read axis independent variable
<additional_axis_variable name="p0" type="int" axis="1">
<attribute name="long_name" value="reference pressure for hybrid sigma coordinate" type="string" />
<attribute name="units" value="Pa" type="string" />
<values>100000</values>
</additional_axis_variable>
-->
</vertical_axis>
</axes>
<variables>
<!--
<parameter id="1" name="geopotential_height">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="long_name" value="geopotential_height" type="string" />
<attribute name="standard_name" value="geopotential_height" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
-->
<!--
<parameter id="2" name="x_wind">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="long_name" value="x_wind" type="string" />
<attribute name="standard_name" value="x_wind" type="string" />
<attribute name="units" value="m/s" type="string" />
<spatial_vector direction="x,longitude" counterpart="y_wind" />
</parameter>
<parameter id="3" name="y_wind">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="long_name" value="y_wind" type="string" />
<attribute name="standard_name" value="y_wind" type="string" />
<attribute name="units" value="m/s" type="string" />
<spatial_vector direction="y,latitude" counterpart="x_wind" />
</parameter>
-->
<!--
<parameter id="10" name="relative_humidity">
<attribute name="scale_factor" value="0.01" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="long_name" value="relative_humidity" type="string" />
<attribute name="standard_name" value="relative_humidity" type="string" />
<attribute name="units" value="1" type="string" />
</parameter>
-->
<parameter id="17,2,1000" name="precipitation_amount" type="float">
<attribute name="_FillValue" value="-32767" type="float" />
<attribute name="long_name" value="precipitation_amount" type="string" />
<attribute name="standard_name" value="precipitation_amount" type="string" />
<attribute name="units" value="kg/m2" type="string" />
</parameter>
<!--
<parameter id="18" name="air_potential_temperature">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="long_name" value="air_potential_temperature" type="string" />
<attribute name="standard_name" value="air_potential_temperature" type="string" />
<attribute name="units" value="K" type="string" />
</parameter>
```

```
-->
<parameter id="25,2,1000" name="cloud_area_fraction">
<attribute name="scale_factor" value="0.01" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="cloud_area_fraction" type="string" />
<attribute name="standard_name" value="cloud_area_fraction" type="string" />
<attribute name="units" value="1" type="string" />
</parameter>
<parameter id="31,2,1000" name="air_temperature">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point height: p" type="string" />
<attribute name="long_name" value="air_temperature" type="string" />
<attribute name="standard_name" value="air_temperature" type="string" />
<attribute name="units" value="K" type="string" />
</parameter>
<parameter id="32,2,1000" name="relative_humidity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="scale_factor" value="0.01" type="float" />
<attribute name="cell_methods" value="time: point height: 2m" type="string" />
<attribute name="long_name" value="relative_humidity" type="string" />
<attribute name="standard_name" value="relative_humidity" type="string" />
<attribute name="units" value="1" type="string" />
</parameter>
<parameter id="33,2,1000" name="x_wind_10m" type="float">
<attribute name="_FillValue" value="9.9692099683868690e+36f" type="float" />
<attribute name="cell_methods" value="time: point height: 10m" type="string" />
<attribute name="long_name" value="x_wind_10m" type="string" />
<attribute name="metno_name" value="x_wind_10m" type="string" />
<attribute name="units" value="m s-1" type="string" />
<spatial_vector direction="x,longitude" counterpart="y_wind_10m" />
</parameter>
<parameter id="34,2,1000" name="y_wind_10m" type="float">
<attribute name="_FillValue" value="9.9692099683868690e+36f" type="float" />
<attribute name="cell_methods" value="time: point height: 10m" type="string" />
<attribute name="long_name" value="y_wind_10m" type="string" />
<attribute name="standard_name" value="y_wind_10m" type="string" />
<attribute name="units" value="m s-1" type="string" />
<spatial_vector direction="y,latitude" counterpart="x_wind_10m" />
</parameter>
<!-- change in scale_factor
<parameter id="36,2,1000" name="accumulated_surface_upward_sensible_heat_flux">
<attribute name="long_name" value="accumulated_surface_upward_sensible_heat_flux" type="string" />
<attribute name="metno_name" value="accumulated_surface_upward_sensible_heat_flux" type="string" />
<attribute name="units" value="kJ/m2" type="string" />
</parameter>
<parameter id="37,2,1000" name="accumulated_surface_upward_latent_heat_flux">
<attribute name="long_name" value="accumulated_surface_upward_latent_heat_flux" type="string" />
<attribute name="metno_name" value="accumulated_surface_upward_latent_heat_flux" type="string" />
<attribute name="units" value="kJ/m2" type="string" />
</parameter>
-->
<parameter id="39" name="cloud_area_fraction_in_atmosphere_layer">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="long_name" value="cloud_area_fraction_in_atmosphere_layer" type="string" />
<attribute name="standard_name" value="cloud_area_fraction_in_atmosphere_layer" type="string" />
<attribute name="scale_factor" value="0.01" type="float" />
<attribute name="units" value="1" type="string" />
</parameter>
<parameter id="58,2,1000" name="sea_level_pressure">
<attribute name="scale_factor" value="100." type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="air_pressure_at_sea_level" type="string" />
<attribute name="standard_name" value="air_pressure_at_sea_level" type="string" />
<attribute name="units" value="Pa" type="string" />
</parameter>
```

```
<parameter id="66,2,1000" name="surface_snow_sickness">
<attribute name="long_name" value="surface_snow_sickness" type="string" />
<attribute name="standard_name" value="surface_snow_sickness" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="101,2,1000" name="altitude" type="short">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="long_name" value="altitude" type="string" />
<attribute name="standard_name" value="altitude" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="103,2,1000" name="sea_surface_temperature">
<attribute name="long_name" value="sea_surface_temperature" type="string" />
<attribute name="metno_name" value="sea_surface_temperature" type="string" />
<attribute name="units" value="K" type="string" />
</parameter>
<parameter id="191,2,1000" name="land_ice_area_fraction">
<attribute name="long_name" value="land_ice_area_fraction" type="string" />
<attribute name="metno_name" value="land_ice_area_fraction" type="string" />
<attribute name="scale_factor" value="0.01" type="float" />
<attribute name="units" value="1" type="string" />
</parameter>
<parameter id="200,3,0" name="significant_wave_height">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_wave_significant_height" type="string" />
<attribute name="standard_name" value="sea_surface_wave_significant_height" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="201,3,0" name="peak_wave_period">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_wave_peak_period" type="string" />
<attribute name="units" value="s" type="string" />
</parameter>
<parameter id="202,3,0" name="significant_wave_period">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_wave_significant_period" type="string" />
<attribute name="units" value="s" type="string" />
</parameter>
<parameter id="203,3,0" name="peak_wave_direction">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_wave_peak_to_direction" type="string" />
<attribute name="units" value="degree" type="string" />
</parameter>
<parameter id="204,3,0" name="wave_direction">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_wave_to_direction" type="string" />
<attribute name="standard_name" value="sea_surface_wave_to_direction" type="string" />
<attribute name="units" value="degree" type="string" />
</parameter>
<parameter id="210,3,0" name="significant_wind_wave_height">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_wind_wave_significant_height" type="string" />
<attribute name="standard_name" value="sea_surface_wind_wave_significant_height" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="213,3,0" name="sea_surface_wind_wave_period">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_wind_wave_period" type="string" />
<attribute name="standard_name" value="sea_surface_wind_wave_period" type="string" />
<attribute name="units" value="s" type="string" />
```

```
</parameter>
<parameter id="214,3,0" name="sea_surface_wind_wave_to_direction">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_wind_wave_to_direction" type="string" />
<attribute name="standard_name" value="sea_surface_wind_wave_to_direction" type="string" />
<attribute name="units" value="degree" type="string" />
</parameter>
<parameter id="220,3,0" name="significant_swell_wave_height">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_swell_wave_significant_height" type="string" />
<attribute name="standard_name" value="sea_surface_swell_wave_significant_height" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="223,3,0" name="sea_surface_swell_wave_period">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_swell_wave_period" type="string" />
<attribute name="standard_name" value="sea_surface_swell_wave_period" type="string" />
<attribute name="units" value="s" type="string" />
</parameter>
<parameter id="224,3,0" name="sea_surface_swell_wave_to_direction">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_swell_wave_to_direction" type="string" />
<attribute name="standard_name" value="sea_surface_swell_wave_to_direction" type="string" />
<attribute name="units" value="degree" type="string" />
</parameter>
<parameter id="291,3,0" name="x_stokes_drift">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="x_sea_wave_stokes_drift" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="292,3,0" name="y_stokes_drift">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="y_sea_wave_stokes_drift" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="301,8,0" name="sea_surface_height">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_height_above_geoid" type="string" />
<attribute name="standard_name" value="sea_surface_height_above_geoid" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="302,5,0" name="x_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_x_velocity" type="string" />
<attribute name="standard_name" value="sea_water_x_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="302,6,0" name="x_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_x_velocity" type="string" />
<attribute name="standard_name" value="sea_water_x_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="302,7,0" name="x_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_x_velocity" type="string" />
<attribute name="standard_name" value="sea_water_x_velocity" type="string" />
```

```
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="302,8,0" name="x_btrop_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="barotropic_sea_water_x_velocity" type="string" />
<attribute name="standard_name" value="sea_water_x_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="303,5,0" name="y_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_y_velocity" type="string" />
<attribute name="standard_name" value="sea_water_y_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="303,6,0" name="y_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_y_velocity" type="string" />
<attribute name="standard_name" value="sea_water_y_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="303,7,0" name="y_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_y_velocity" type="string" />
<attribute name="standard_name" value="sea_water_y_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="303,8,0" name="y_btrop_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="barotropic_sea_water_y_velocity" type="string" />
<attribute name="standard_name" value="sea_water_y_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="304,5,0" name="z_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="upward_sea_water_velocity" type="string" />
<attribute name="standard_name" value="upward_sea_water_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="304,7,0" name="z_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="upward_sea_water_velocity" type="string" />
<attribute name="standard_name" value="upward_sea_water_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="305,5,0" name="sigma_t">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_sigma_t" type="string" />
<attribute name="standard_name" value="sea_water_sigma_t" type="string" />
<attribute name="units" value="kg m-3" type="string" />
</parameter>
<parameter id="305,6,0" name="sigma_t">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_sigma_t" type="string" />
<attribute name="standard_name" value="sea_water_sigma_t" type="string" />
<attribute name="units" value="kg m-3" type="string" />
</parameter>
<parameter id="305,7,0" name="sigma_t">
<attribute name="_FillValue" value="-32767" type="short" />
```

```
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_sigma_t" type="string" />
<attribute name="standard_name" value="sea_water_sigma_t" type="string" />
<attribute name="units" value="kg m-3" type="string" />
</parameter>
<parameter id="306,6,0" name="layer_thickness">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_layer_thickness" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="306,7,0" name="layer_thickness">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_layer_thickness" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="307,5,0" name="salinity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_salinity" type="string" />
<attribute name="standard_name" value="sea_water_salinity" type="string" />
<attribute name="units" value="1e-3" type="string" />
</parameter>
<parameter id="307,6,0" name="salinity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_salinity" type="string" />
<attribute name="standard_name" value="sea_water_salinity" type="string" />
<attribute name="units" value="1e-3" type="string" />
</parameter>
<parameter id="307,7,0" name="salinity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_salinity" type="string" />
<attribute name="standard_name" value="sea_water_salinity" type="string" />
<attribute name="units" value="1e-3" type="string" />
</parameter>
<parameter id="307,8,0" name="salinity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_salinity" type="string" />
<attribute name="standard_name" value="sea_surface_salinity" type="string" />
<attribute name="units" value="1e-3" type="string" />
</parameter>
<parameter id="308,5,0" name="sea_temperature">
<attribute name="add_offset" value="273.15" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_temperature" type="string" />
<attribute name="standard_name" value="sea_water_temperature" type="string" />
<attribute name="units" value="K" type="string" />
</parameter>
<parameter id="308,6,0" name="sea_temperature">
<attribute name="add_offset" value="273.15" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_temperature" type="string" />
<attribute name="standard_name" value="sea_water_temperature" type="string" />
<attribute name="units" value="K" type="string" />
</parameter>
```

```
<parameter id="308,7,0" name="sea_temperature">
<attribute name="add_offset" value="273.15" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_temperature" type="string" />
<attribute name="standard_name" value="sea_water_temperature" type="string" />
<attribute name="units" value="K" type="string" />
</parameter>
<parameter id="308,8,0" name="sea_surface_temperature">
<attribute name="add_offset" value="273.15" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_surface_temperature" type="string" />
<attribute name="standard_name" value="sea_surface_temperature" type="string" />
<attribute name="units" value="K" type="string" />
</parameter>
<parameter id="312,5,0" name="TKE">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_turbulent_kinetic_energy" type="string" />
<attribute name="units" value="m2 s-2" type="string" />
</parameter>
<parameter id="312,6,0" name="TKE">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_turbulent_kinetic_energy" type="string" />
<attribute name="units" value="m2 s-2" type="string" />
</parameter>
<parameter id="312,7,0" name="TKE">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_water_turbulent_kinetic_energy" type="string" />
<attribute name="units" value="m2 s-2" type="string" />
</parameter>
<parameter id="331,8,0" name="sea_surface_height">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_surface_height_above_geoid" type="string" />
<attribute name="standard_name" value="sea_surface_height_above_geoid" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="332,5,0" name="x_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_x_velocity" type="string" />
<attribute name="standard_name" value="sea_water_x_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="332,6,0" name="x_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_x_velocity" type="string" />
<attribute name="standard_name" value="sea_water_x_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="332,7,0" name="x_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_x_velocity" type="string" />
<attribute name="standard_name" value="sea_water_x_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="332,8,0" name="x_btrop_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
```

```
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="barotropic_sea_water_x_velocity" type="string" />
<attribute name="standard_name" value="sea_water_x_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="333,5,0" name="y_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_y_velocity" type="string" />
<attribute name="standard_name" value="sea_water_y_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="333,6,0" name="y_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_y_velocity" type="string" />
<attribute name="standard_name" value="sea_water_y_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="333,7,0" name="y_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_y_velocity" type="string" />
<attribute name="standard_name" value="sea_water_y_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="333,8,0" name="y_btrop_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="barotropic_sea_water_y_velocity" type="string" />
<attribute name="standard_name" value="sea_water_y_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="334,5,0" name="z_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="upward_sea_water_velocity" type="string" />
<attribute name="standard_name" value="upward_sea_water_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="334,7,0" name="z_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="upward_sea_water_velocity" type="string" />
<attribute name="standard_name" value="upward_sea_water_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="335,5,0" name="sigma_t">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_sigma_t" type="string" />
<attribute name="standard_name" value="sea_water_sigma_t" type="string" />
<attribute name="units" value="kg m-3" type="string" />
</parameter>
<parameter id="335,6,0" name="sigma_t">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_sigma_t" type="string" />
<attribute name="standard_name" value="sea_water_sigma_t" type="string" />
<attribute name="units" value="kg m-3" type="string" />
</parameter>
<parameter id="335,7,0" name="sigma_t">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_sigma_t" type="string" />
<attribute name="standard_name" value="sea_water_sigma_t" type="string" />
<attribute name="units" value="kg m-3" type="string" />
```

```
</parameter>
<parameter id="336,6,0" name="layer_thickness">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_layer_thickness" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="336,7,0" name="layer_thickness">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_layer_thickness" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="337,5,0" name="salinity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_salinity" type="string" />
<attribute name="standard_name" value="sea_water_salinity" type="string" />
<attribute name="units" value="1e-3" type="string" />
</parameter>
<parameter id="337,6,0" name="salinity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_salinity" type="string" />
<attribute name="standard_name" value="sea_water_salinity" type="string" />
<attribute name="units" value="1e-3" type="string" />
</parameter>
<parameter id="337,7,0" name="salinity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_salinity" type="string" />
<attribute name="standard_name" value="sea_water_salinity" type="string" />
<attribute name="units" value="1e-3" type="string" />
</parameter>
<parameter id="337,8,0" name="salinity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_surface_salinity" type="string" />
<attribute name="standard_name" value="sea_surface_salinity" type="string" />
<attribute name="units" value="1e-3" type="string" />
</parameter>
<parameter id="338,5,0" name="temperature">
<attribute name="add_offset" value="273.15" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_temperature" type="string" />
<attribute name="standard_name" value="sea_water_temperature" type="string" />
<attribute name="units" value="K" type="string" />
</parameter>
<parameter id="338,6,0" name="temperature">
<attribute name="add_offset" value="273.15" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_water_temperature" type="string" />
<attribute name="standard_name" value="sea_water_temperature" type="string" />
<attribute name="units" value="K" type="string" />
</parameter>
<parameter id="338,7,0" name="temperature">
<attribute name="add_offset" value="273.15" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
```

```
<attribute name="long_name" value="sea_water_temperature" type="string" />
<attribute name="standard_name" value="sea_water_temperature" type="string" />
<attribute name="units" value="K" type="string" />
</parameter>
<parameter id="338,8,0" name="temperature">
<attribute name="add_offset" value="273.15" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: mean" type="string" />
<attribute name="long_name" value="sea_surface_temperature" type="string" />
<attribute name="standard_name" value="sea_surface_temperature" type="string" />
<attribute name="units" value="K" type="string" />
</parameter>
<parameter id="339,2,0" name="total_precipitation_rate">
<attribute name="valid_min" value="0" type="short" />
<attribute name="scale_factor" value="4.630e-8" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="lwe_precipitation_rate" type="string" />
<attribute name="standard_name" value="lwe_precipitation_rate" type="string" />
<attribute name="units" value="m s-1" type="string" />
<attribute name="unitsNOTE" value="assumes met.no FELT values are mm/6h" type="string" />
</parameter>
<parameter id="340,8,0" name="sea_ice_concentration">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_ice_area_fraction" type="string" />
<attribute name="standard_name" value="sea_ice_area_fraction" type="string" />
<attribute name="units" value="1e-2" type="string" />
</parameter>
<parameter id="341,8,0" name="sea_ice_thickness">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_ice_thickness" type="string" />
<attribute name="standard_name" value="sea_ice_thickness" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="342,6,0" name="x_ice_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_ice_x_velocity" type="string" />
<attribute name="standard_name" value="sea_ice_x_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="342,8,0" name="x_ice_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_ice_x_velocity" type="string" />
<attribute name="standard_name" value="sea_ice_x_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="343,6,0" name="y_ice_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_ice_y_velocity" type="string" />
<attribute name="standard_name" value="sea_ice_y_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="343,8,0" name="y_ice_velocity">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="sea_ice_y_velocity" type="string" />
<attribute name="standard_name" value="sea_ice_y_velocity" type="string" />
<attribute name="units" value="m s-1" type="string" />
</parameter>
<parameter id="351,8,0" name="bottom_topography">
```

```
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="long_name" value="sea_floor_depth_below_geoid" type="string" />
<attribute name="standard_name" value="sea_floor_depth_below_geoid" type="string" />
<attribute name="units" value="m" type="string" />
</parameter>
<parameter id="371,5,0" name="nitrate">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="scale_factor" value="6.969e-08" type="float" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="moles_of_nitrate_per_unit_mass_in_sea_water" type="string" />
<attribute name="standard_name" value="moles_of_nitrate_per_unit_mass_in_sea_water" type="string" />
<attribute name="units" value="mol kg-1" type="string" />
</parameter>
<parameter id="372,5,0" name="phosphate">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="scale_factor" value="3.147e-08" type="float" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="moles_of_phosphate_per_unit_mass_in_sea_water" type="string" />
<attribute name="standard_name" value="moles_of_phosphate_per_unit_mass_in_sea_water" type="string" />
<attribute name="units" value="mol kg-1" type="string" />
</parameter>
<parameter id="373,5,0" name="silicate">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="scale_factor" value="3.484e-08" type="float" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="moles_of_silicate_per_unit_mass_in_sea_water" type="string" />
<attribute name="standard_name" value="moles_of_silicate_per_unit_mass_in_sea_water" type="string" />
<attribute name="units" value="mol kg-1" type="string" />
</parameter>
<parameter id="374,5,0" name="detritus">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="scale_factor" value="7.143e-05" type="float" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="concentration_of_detritius_in_sea_water" type="string" />
<attribute name="standard_name" value="mole_concentration_of_organic_detritus_in_sea_water_expressed_as_ni
<attribute name="units" value="mol m-3" type="string" />
<attribute name="units_assumption" value="units based on a density of 1025 kg m-3, not actual density" typ
</parameter>
<parameter id="375,5,0" name="diatoms">
<attribute name="valid_min" value="0" type="short" />
<attribute name="scale_factor" value="7.143e-05" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="concentration_of_diatoms_in_sea_water" type="string" />
<attribute name="standard_name" value="mole_concentration_of_diatoms_in_sea_water_expressed_as_nitrogen" t
<attribute name="units" value="mol m-3" type="string" />
<attribute name="units_assumption" value="units based on a density of 1025 kg m-3, not actual density" typ
</parameter>
<parameter id="376,5,0" name="flagellates">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="valid_min" value="0" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="concentration_of_flagellates_in_sea_water" type="string" />
<attribute name="units" value="mgN m-3" type="string" />
</parameter>
<parameter id="377,5,0" name="oxygen">
<attribute name="valid_min" value="0" type="short" />
<attribute name="scale_factor" value="1.e-06" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="mass_concentration_of_oxygen_in_sea_water" type="string" />
<attribute name="standard_name" value="mass_concentration_of_oxygen_in_sea_water" type="string" />
```

```
<attribute name="units" value="kg m-3" type="string" />
</parameter>
<parameter id="378,5,0" name="silica">
<attribute name="valid_min" value="0" type="short" />
<attribute name="scale_factor" value="3.484e-08" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="moles_of_silica_per_unit_mass_in_sea_water" type="string" />
<attribute name="units" value="mol kg-1" type="string" />
</parameter>
<parameter id="379,5,0" name="sediment_type_1">
<attribute name="valid_min" value="0" type="short" />
<attribute name="scale_factor" value="1.e-06" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="mass_concentration_of_fast_sinking_material_in_sea_water" type="string"
<attribute name="units" value="mgN m-3" type="string" />
</parameter>
<parameter id="380,5,0" name="sediment_type_2">
<attribute name="valid_min" value="0" type="short" />
<attribute name="scale_factor" value="1.e-06" type="float" />
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="cell_methods" value="time: point" type="string" />
<attribute name="long_name" value="mass_concentration_of_slow_sinking_material_in_sea_water" type="string"
<attribute name="units" value="mgN m-3" type="string" />
</parameter>
</variables>
</cdm_felt_config>
```

Chapter 4

gribWriter Configuration

4.1 gribWriter Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<cdm_gribwriter_config>
<global_attributes>
<!-- type can be double, long, string -->
<attribute name="identificationOfOriginatingGeneratingCentre" value="98" type="long" /> <!-- currently usi
<!-- use for grib1 only attributes -->
<!--  <g1attribute name="" value="" type="" /> -->
<!-- use for grib2 only attributes -->
<g2attribute name="shapeOfTheEarth" value="8" type="long" /> <!-- spherical r=6,371,229.0m -->
<g2attribute name="typeOfPacking" value="grid_simple" type="string" /> <!-- jpeg2000 -->
</global_attributes>
<axes>
<!-- no vertical axis -->
<vertical_axis standard_name="">
   <!-- using height in meter, 0 -->
   <grib1 id="105" value="0" type="short"/>
   <grib2 id="103" value="0" type="short"/>
</vertical_axis>
<vertical_axis unitCompatibleTo="m">
   <!-- no standard_name for height/depth, defined only by positive=up/down -->
   <grib1 id="105" units="m" type="short"/>
   <grib2 id="103" units="m" type="double"/>
</vertical_axis>
<vertical_axis unitCompatibleTo="Pa">
   <!-- no standard_name for pressure coord, defined only by unit of hPa -->
   <grib1 id="100" units="hPa" type="short"/>
   <grib2 id="100" units="Pa" type="double"/>
</vertical_axis>
<vertical_axis standard_name="atmosphere_sigma_coordinate">
<grib1 id="107" scale_factor="1e-4" add_offset="0" type="short" />
<grib2 id="104" type="double" />
</vertical_axis>
</axes>
<variables>
<!-- translation section for parameter names/standard_names -->
<parameter standard_name="altitude">
    <grib1 parameterNumber="8" codeTable="128" units="m"/>
    <grib2 discipline="0" parameterCategory="3" parameterNumber="6" units="m"/>
</parameter>
<parameter standard_name="sea_surface_temperature">
    <grib1 parameterNumber="80" codeTable="128" units="K"/>
    <grib2 discipline="10" parameterCategory="3" parameterNumber="0" units="K"/>
</parameter>
<parameter standard_name="precipitation_amount">
    <grib1 parameterNumber="61" codeTable="128" units="kg/m2"/>
    <grib2 discipline="0" parameterCategory="1" parameterNumber="8" units="kg/m2"/>
</parameter>
<parameter standard_name="cloud_area_fraction">
<!-- total -->
    <grib1 parameterNumber="164" codeTable="128" units="1/100"/>
</parameter>
<parameter standard_name="cloud_area_fraction_in_atmosphere_layer" level="10000">
<!-- fog -->
    <grib1 parameterNumber="248" codeTable="128" units="1/100"/>
</parameter>
<parameter standard_name="cloud_area_fraction_in_atmosphere_layer" level="8500">
<!-- low clouds  -->
    <grib1 parameterNumber="186" codeTable="128" units="1/100"/>
</parameter>
<parameter standard_name="cloud_area_fraction_in_atmosphere_layer" level="5000">
<!--  medium clouds  -->
    <grib1 parameterNumber="187" codeTable="128" units="1/100"/>
</parameter>
<parameter standard_name="cloud_area_fraction_in_atmosphere_layer" level="3000">
<!-- high clouds -->
```

```
    <grib1 parameterNumber="188" codeTable="128" units="1/100"/>
</parameter>
</variables>
</cdm_gribwriter_config>
```

Chapter 5

netcdfWriter Configuration

5.1 netcdfWriter Configuration

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.
- Different units are required for special usages.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cdm_ncwriter_config SYSTEM "cdmWriterConfig.dtd">
<cdm_ncwriter_config>

<!-- all operations below can (and must) be joined if the -->
<!-- same attribute/dimension/variable is concerned -->
<!-- i.e. a <variable name="x"... may not occur twice!!! -->

<!-- add/reassign a new global attribute -->
<attribute name="max_time" value="2008-05-28" type="string" />

<!-- add/reassign a new variable attribute -->
<variable name="air_temperature">
<attribute name="standard_name" value="temperature" type="string" />
</variable>

<!-- change the type of a variable -->
<variable name="precipitation_amount" type="short">
<attribute name="_FillValue" value="-32767" type="short" />
<attribute name="scale_factor" value="0.001" type="float" />
<attribute name="add_offset" value="0" type="double" />
</variable>

<!-- rename a variable -->
<variable name="sea_level_pressure" newname="sea_pressure" />

<!-- rename a dimension -->
<dimension name="x" newname="x_c" />

<!--  change units from m to km-->
<!--  make sure to put the type to change the data, too!!! -->
<variable name="x" type="double">
<attribute name="units" value="km" type="string" />
</variable>

<!-- rename a attribute -->
<attribute name="min_time" newname="minimum_time" />
</cdm_ncwriter_config>
```

Chapter 6

Namespace Index

6.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 7

Class Index

7.1 Class Hierarchy

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

Chapter 8

Class Index

8.1 Class List

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

Chapter 9

File Index

9.1 File List

Here is a list of all files with brief descriptions:

Chapter 10

Namespace Documentation

10.1 MetNoFelt Namespace Reference

Classes

- struct ShortPairLess
- class Felt_Array

  encapsulate parameters of a felt file

- class Felt_File

  Felt File access.

- class Felt_File_Error
- class FeltParameters

Typedefs

- typedef **set**< **pair**< short, short >, ShortPairLess > ShortPairSet
- typedef **map**< **pair**< short, short >, short, ShortPairLess > ShortPairMap

Functions

- time_t index16toTime (const boost::array< short, 16 > &idx)
- **pair**< short, short > index16toLevelPair (const boost::array< short, 16 > &idx)
- **std::string** getProjString (int gridType, const boost::array< float, 6 > &gridParameters) throw (Felt_File_Error)
- const int ANY_VALUE ()
- const **std::string** & UNDEFINED ()
- const boost::array< short, 16 > & ANY_ARRAY ()
- const boost::array< short, 20 > & ANY_ARRAY20 ()

10.1.1 Typedef Documentation

10.1.1.1 typedef map<pair<short,short>, short, ShortPairLess > MetNoFelt::ShortPairMap

map<pair<short,short>, short> with comparator

10.1.1.2 typedef set<pair<short,short>, ShortPairLess > MetNoFelt::ShortPairSet

set<pair<short,short> > with comparator

10.1.2 Function Documentation

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

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

10.1.2.3 const int MetNoFelt::ANY_VALUE () [inline]

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

10.1.2.5 pair<short, short> MetNoFelt::index16toLevelPair (const boost::array< short, 16 > & *idx*)

convert the 16-short header to a levelPair

10.1.2.6 time_t MetNoFelt::index16toTime (const boost::array< short, 16 > & *idx*)

convert the 16-short header to a time

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

10.2 MetNoFimex Namespace Reference

Classes

- class CachedInterpolation
- class CachedVectorReprojection
- class CDM

  Data structure of the Common Data Model.

- class CDMAttribute
- class CDMDimension
- class CDMException
- class CDMExtractor
- class CDMInterpolator
- class CDMNamedEntity
- struct CDMNameCompare
- class CDMNameEqual
- class CDMReader

  Basic interface for CDM reading and manipulation classes.

- class CDMTimeInterpolator
- class CDMVariable
- class CDMWriter
- class Data
- class DataImpl
- class DataTypeChanger
- class FeltCDMReader
- class GribApiCDMWriter
- class GribApiCDMWriter_Impl1
- class GribApiCDMWriter_Impl2
- class GribApiCDMWriter_ImplAbstract
- class Logger
- class NetCDF_CDMWriter
- class NetCDF_CF10_CDMReader
- class Null_CDMWriter
- class ReplaceStringObject
- class ReplaceStringTimeObject
- class TimeLevelDataSliceFetcher

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

- class TimeSpec
- class FimexTime
- class TimeUnit
- class UnitException
- class Units
- class XMLDoc

Typedefs

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

Enumerations

- enum CDMDataType {
  CDM_NAT = 0, CDM_CHAR, CDM_SHORT, CDM_INT,
  CDM_FLOAT, CDM_DOUBLE, CDM_STRING }

Functions

- **std::vector**< CDMAttribute > projStringToAttributes (**std::string** projStr)

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

- **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::shared_ptr< Data > createData (CDMDataType datatype, size_t length) throw (CDMException)

  create a Data-pointer of the datatype

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

  create a dataslice from another Data object

- template<typename T1, typename T2>
  boost::shared_array< T1 > duplicateArrayType (const boost::shared_array< T2 > &inData, long length)

  create a new shared array with a different type using static_cast

- template<typename T1, typename T2>
  const boost::shared_array< T1 > constConvertArrayType (const boost::shared_array< T2 > &inData, long length)

  return a shared array of this data, possibly pointer to internal data

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

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

- template<typename C>
  void recursiveCopyMultiDimData (C **orgData, C **newData, const **std::vector**< size_t > &orgDimSize, const **std::vector**< size_t > &orgSliceSize, const **std::vector**< size_t > &newStart, const **std::vector**< size_t > &newSize, size_t currentDim)
- template<typename T1, typename T2>
  boost::shared_array< T1 > convertArrayType (const boost::shared_array< T2 > &inData, size_t length, double oldFill, double oldScale, double oldOffset, double newFill, double newScale, double newOffset)
- Logger::LogLevel defaultLogLevel ()

- void defaultLogLevel (Logger::LogLevel)
- LoggerPtr getLogger (const **std::string** &className)
- NcType cdmDataType2ncType (CDMDataType dt)
- CDMDataType ncType2cdmDataType (NcType dt)
- boost::shared_ptr< Data > ncValues2Data (NcValues *values, NcType dt, size_t length)
- **std::ostream** & operator<< (**std::ostream** &out, const FimexTime &fTime)
- FimexTime string2FimexTime (const **std::string** &str) throw (CDMException)
- void handleUdUnitError (int unitErrCode, const **std::string** &message="") throw (UnitException)
- int round (double num)
- **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<typename T>
  T string2type (**std::string** s)
- **std::string** getXmlProp (const xmlNodePtr node, const **std::string** &attrName)
- **std::string** getXmlName (const xmlNodePtr node)

10.2.1 Typedef Documentation

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

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

10.2.2 Enumeration Type Documentation

10.2.2.1 enum MetNoFimex::CDMDataType

Enumerator:

CDM_NAT
CDM_CHAR
CDM_SHORT
CDM_INT
CDM_FLOAT
CDM_DOUBLE
CDM_STRING

10.2.3 Function Documentation

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

convert attributes of a projection-variable to a projString

Parameters:

attrs attributes of the projection variable

Returns:

proj4 string

10.2.3.2 NcType MetNoFimex::cdmDataType2ncType (CDMDataType *dt*)

conversion from CDMDataType to NcType

10.2.3.3 template<typename T1, typename T2> const boost::shared_array< T1 > MetNoFimex::constConvertArrayType (const boost::shared_array< T2 > & *inData*, long *length*) [inline]

return a shared array of this data, possibly pointer to internal data

Parameters:

inData original data

length length of original data array

10.2.3.4 template<typename T1, typename T2> boost::shared_array<T1> MetNoFimex::convertArrayType (const boost::shared_array< T2 > & *inData*, size_t *length*, double *oldFill*, double *oldScale*, double *oldOffset*, double *newFill*, double *newScale*, double *newOffset*) [inline]

References fill().

10.2.3.5 template<class InputIterator> boost::shared_ptr< Data > MetNoFimex::createData (CDMDataType *datatype*, size_t *length*, InputIterator *first*, InputIterator *last*) throw (CDMException) [inline]

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

Parameters:

datatype

size_t length of the data array

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_NAT, and CDM_SHORT.

10.2.3.6 boost::shared_ptr<Data> MetNoFimex::createData (CDMDataType *datatype*, size_t *length*) throw (CDMException)

create a Data-pointer of the datatype

Parameters:

datatype

size_t length of the data array

Returns:

Base-Class ptr of the DataImpl belonging to the datatype

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

create a dataslice from another Data object

Parameters:

datatype

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

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

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

10.2.3.10 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.

10.2.3.11 template<typename T1, typename T2> boost::shared_array< T1 > MetNoFimex::duplicateArrayType (const boost::shared_array< T2 > & *inData*, long *length*) `[inline]`

create a new shared array with a different type using static_cast

Parameters:

inData original data

length length of original data array

10.2.3.12 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

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

a memory-save form of xmlGetName

Returns:

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

10.2.3.14 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

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

10.2.3.16 CDMDataType MetNoFimex::ncType2cdmDataType (NcType *dt*)

conversion from NcType to CDMDataType

10.2.3.17 boost::shared_ptr<Data> MetNoFimex::ncValues2Data (NcValues * *values*, NcType *dt*, size_t *length*)

convert ncValues to a Data pointer

Warning:

: the data belonging to values will be freed within this function or with the shared_array. Do not free the values otherwise!

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

10.2.3.19 std::vector<CDMAttribute> MetNoFimex::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=?

10.2.3.20 template<typename C> void MetNoFimex::recursiveCopyMultiDimData (C ** *orgData*, C ** *newData*, const std::vector< size_t > & *orgDimSize*, const std::vector< size_t > & *orgSliceSize*, const std::vector< size_t > & *newStart*, const std::vector< size_t > & *newSize*, size_t *currentDim*) [inline]

recursively copy data by moving the newData and orgData pointers forward and copy the data at the current position

it's assumed that the first dim in the vector is the fastest moving (fortran like)

Parameters:

orgData pointer to the current postion of the original array

newData pointer to the current position of the new array the original dimensions of orgData helper-array with orgSliceSize[0] = 1; orgSliceSize[n] = orgDimSize[n] * orgSliceSize[n-1] the start positions in the new data the dimensions of the newData the dimension currently under work, should be between (orgData.size()-1) and 0

Referenced by MetNoFimex::DataImpl< C >::slice().

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

Round a double to integer.

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

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

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

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

convert a string to lowercase

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

10.2.3.26 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

Returns:

vector of tokens

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

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

References std::basic_ostringstream< _CharT, _Traits, _Alloc >::str().

Referenced by MetNoFimex::DataImpl< C >::setValues(), and MetNoFimex::DataImpl< C >::slice().

Chapter 11

Class Documentation

11.1 MetNoFimex::CachedInterpolation Class Reference

`#include <CachedInterpolation.h>`

Public Member Functions

- CachedInterpolation ()
- 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 ()
- boost::shared_array< float > interpolateValues (boost::shared_array< float > inData, size_t size, size_t &newSize)

11.1.1 Detailed Description

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

11.1.2 Constructor & Destructor Documentation

11.1.2.1 MetNoFimex::CachedInterpolation::CachedInterpolation () `[inline]`

11.1.2.2 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

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

11.1.3 Member Function Documentation

11.1.3.1 boost::shared_array<float> MetNoFimex::CachedInterpolation::interpolateValues (boost::shared_array< float > *inData*, size_t *size*, size_t & *newSize*)

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

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

- include/fimex/CachedInterpolation.h

11.2 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

11.2.1 Constructor & Destructor Documentation

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

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

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

11.2.2 Member Function Documentation

11.2.2.1 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

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

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

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

- include/fimex/CachedVectorReprojection.h

11.3 MetNoFimex::CDM Class Reference

Data structure of the Common Data Model.

`#include <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 ()
- virtual ~CDM ()
- 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 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)
- 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
- 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")

- 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

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

get the projection attributes (as of CF-1.0) 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

11.3.1 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

11.3.2 Member Typedef Documentation

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

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

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

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

11.3.3 Constructor & Destructor Documentation

11.3.3.1 MetNoFimex::CDM::CDM ()

11.3.3.2 virtual MetNoFimex::CDM::~CDM () [virtual]

11.3.4 Member Function Documentation

11.3.4.1 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

11.3.4.2 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

11.3.4.3 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

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

test if variable exists

Parameters:

varName name of variable

11.3.4.5 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:

copies of the attributes matching the request

11.3.4.6 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:

copies of the attributes matching the request

11.3.4.7 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

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

remove a variable and corresponding attributes

Parameters:

variableName the variable to remove

11.3.4.9 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

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

check if the dimension exists

Parameters:

dimName name of the dimension

11.3.4.11 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

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

11.3.4.13 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

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

test if a variable contains the unlimited dim

Returns:

true/false

11.3.4.15 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

11.3.4.16 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

11.3.4.17 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

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

print a xml representation to the stream

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

the namespace for global attributes

11.3.4.20 const DimVec& MetNoFimex::CDM::getDimensions () const [inline]

get the dimension

11.3.4.21 const VarVec& MetNoFimex::CDM::getVariables () const [inline]

get the variables

11.3.4.22 const StrAttrVecMap& MetNoFimex::CDM::getAttributes () const [inline]

get the attributes

Returns:

map of type <variableName <attributeName, attribute>>

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

get the attributes of an variable

Parameters:

varName name of variable

11.3.4.24 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

11.3.4.25 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

11.3.4.26 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

11.3.4.27 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_F

11.3.4.28 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

11.3.4.29 bool MetNoFimex::CDM::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

11.3.4.30 AttrVec MetNoFimex::CDM::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

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

get the x-(lon) axis of the variable

Parameters:

varName name of variable

Returns:

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

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

get the y-(lat) axis of the variable

Parameters:

varName name of variable

Returns:

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

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

detect the latitude and longitude coordinates of the variable

Detect the the latitude and longitude coordinates of the variable, this might be one of the dimensions, or a multi-dimensional field of lat(x,y) lon(x,y) variables

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

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

get the time axis of the variable

Parameters:

varName name of variable

Returns:

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

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

get the vertical axis of the variable

Parameters:

varName name of variable

Returns:

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

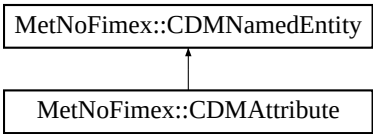
The documentation for this class was generated from the following file:

- include/fimex/CDM.h

11.4 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

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

  retrieve 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

11.4.1 Constructor & Destructor Documentation

11.4.1.1 MetNoFimex::CDMAttribute::CDMAttribute ()

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

create a string attribute

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

create a char attribute with a char array of length 1

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

create a int attribute with a int array of length 1

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

create a short attribute with a short array of length 1

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

create a float attribute with a float array of length 1

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

create a double attribute with a double array of length 1

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

create a attribute with the low level information

11.4.1.9 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

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

11.4.2 Member Function Documentation

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

retrieve the name of the attribute

Implements MetNoFimex::CDMNamedEntity.

11.4.2.2 const std::string MetNoFimex::CDMAttribute::getStringValue () const [inline]

retrieve the stringified value of the attribute

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

retrieve the data-pointer of the attribute

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

set the data for this attribute

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

retrieve the datatype of the attribute

11.4.2.6 void MetNoFimex::CDMAttribute::toXMLStream (std::ostream & *out*) const

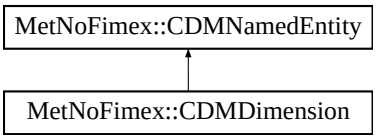
The documentation for this class was generated from the following file:

- include/fimex/CDMAttribute.h

11.5 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
- 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

11.5.1 Constructor & Destructor Documentation

11.5.1.1 MetNoFimex::CDMDimension::CDMDimension ()

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

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

11.5.2 Member Function Documentation

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

Implements MetNoFimex::CDMNamedEntity.

11.5.2.2 size_t MetNoFimex::CDMDimension::getLength () const [inline]

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

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

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

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

print xml representation to stream

Parameters:

out stream to write xml to

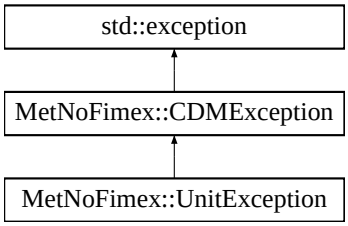
The documentation for this class was generated from the following file:

- include/fimex/CDMDimension.h

11.6 MetNoFimex::CDMException Class Reference

`#include <CDMException.h>`

Inheritance diagram for MetNoFimex::CDMException::



Public Member Functions

- CDMException ()
- CDMException (const **std::string** &msg)
- CDMException (const CDMException &rhs) throw ()
- CDMException & operator= (const CDMException &rhs) throw ()
- virtual ~CDMException () throw ()
- virtual const char * what () const throw ()

11.6.1 Constructor & Destructor Documentation

11.6.1.1 MetNoFimex::CDMException::CDMException () `[inline]`

11.6.1.2 MetNoFimex::CDMException::CDMException (const std::string & *msg*) `[inline, explicit]`

11.6.1.3 MetNoFimex::CDMException::CDMException (const CDMException & *rhs*) throw () `[inline]`

11.6.1.4 virtual MetNoFimex::CDMException::~CDMException () throw () `[inline, virtual]`

11.6.2 Member Function Documentation

11.6.2.1 CDMException& MetNoFimex::CDMException::operator= (const CDMException & *rhs*) throw () `[inline]`

11.6.2.2 virtual const char* MetNoFimex::CDMException::what () const throw () `[inline, virtual]`

Reimplemented from **std::exception**.

References std::basic_string< _CharT, _Traits, _Alloc >::c_str().

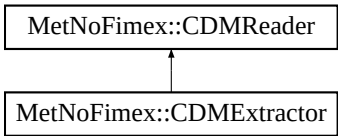
The documentation for this class was generated from the following file:

- include/fimex/CDMException.h

11.7 MetNoFimex::CDMExtractor Class Reference

`#include <CDMExtractor.h>`

Inheritance diagram for MetNoFimex::CDMExtractor::



Public Member Functions

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

  data-reading function to be called from the CDMWriter

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

  Remove a variable from the CDM.

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

  Reduce a dimension of the file.

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

  Reduce a dimension of the file.

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

  change the datatype of the variable

11.7.1 Constructor & Destructor Documentation

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

11.7.1.2 virtual MetNoFimex::CDMExtractor::~CDMExtractor () `[virtual]`

11.7.2 Member Function Documentation

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

data-reading function to be called from the CDMWriter

This function 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

Implements MetNoFimex::CDMReader.

11.7.2.2 virtual void MetNoFimex::CDMExtractor::removeVariable (std::string *variable*) throw (CDMException) `[virtual]`

Remove a variable from the CDM.

Parameters:

name of the variable

Exceptions:

CDMException if variable doesn't exist

11.7.2.3 virtual void MetNoFimex::CDMExtractor::reduceDimension (std::string *dimName*, size_t *start*, size_t *length*) throw (CDMException) `[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

11.7.2.4 virtual void MetNoFimex::CDMExtractor::reduceDimensionStartEnd (std::string *dimName*, size_t *start* = `0`, long *end* = `0`) throw (CDMException) `[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

11.7.2.5 virtual void MetNoFimex::CDMExtractor::changeDataType (std::string *variable*, CDMDataType *datatype*) throw (CDMException) `[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

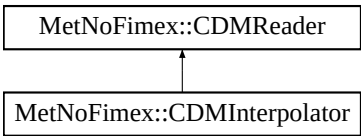
The documentation for this class was generated from the following file:

- include/fimex/CDMExtractor.h

11.8 MetNoFimex::CDMInterpolator Class Reference

`#include <CDMInterpolator.h>`

Inheritance diagram for MetNoFimex::CDMInterpolator::



Public Member Functions

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

  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) throw (CDMException)
- 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

11.8.1 Constructor & Destructor Documentation

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

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

11.8.2 Member Function Documentation

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

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

11.8.2.2 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*) throw (CDMException) `[virtual]`

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

Parameters:

method Interpolation method

11.8.2.3 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

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

Returns:

the name used for latitude in the automatic coordinate generation

11.8.2.5 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

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

Returns:

the name used for longitude in the automatic coordinate generation

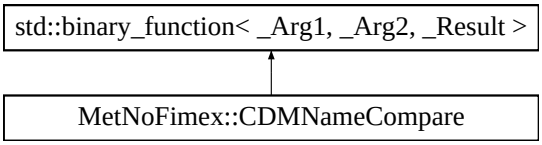
The documentation for this class was generated from the following file:

- include/fimex/CDMInterpolator.h

11.9 MetNoFimex::CDMNameCompare Struct Reference

`#include <CDMNamedEntity.h>`

Inheritance diagram for MetNoFimex::CDMNameCompare::



Public Member Functions

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

11.9.1 Detailed Description

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

11.9.2 Member Function Documentation

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

References std::basic_string< _CharT, _Traits, _Alloc >::compare(), and MetNoFimex::CDMNamedEntity::getName().

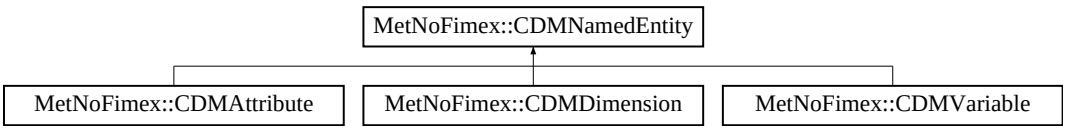
The documentation for this struct was generated from the following file:

- include/fimex/CDMNamedEntity.h

11.10 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

11.10.1 Detailed Description

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

11.10.2 Constructor & Destructor Documentation

11.10.2.1 virtual MetNoFimex::CDMNamedEntity::~CDMNamedEntity () `[pure virtual]`

11.10.3 Member Function Documentation

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

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

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

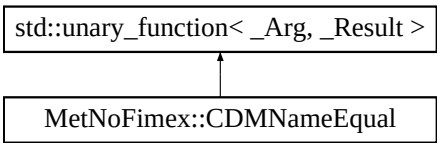
The documentation for this class was generated from the following file:

- include/fimex/CDMNamedEntity.h

11.11 MetNoFimex::CDMNameEqual Class Reference

`#include <CDMNamedEntity.h>`

Inheritance diagram for MetNoFimex::CDMNameEqual::



Public Member Functions

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

11.11.1 Detailed Description

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

11.11.2 Constructor & Destructor Documentation

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

11.11.2.2 MetNoFimex::CDMNameEqual::CDMNameEqual (CDMNamedEntity & *entity*) `[inline, explicit]`

11.11.2.3 MetNoFimex::CDMNameEqual::~CDMNameEqual () `[inline]`

11.11.3 Member Function Documentation

11.11.3.1 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

11.12 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 const boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos)=0 throw (CDMException)

  data-reading function to be called from the CDMWriter

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

  data-reading function to be called from the CDMWriter

Protected Member Functions

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

Protected Attributes

- CDM cdm

11.12.1 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

11.12.2 Constructor & Destructor Documentation

11.12.2.1 MetNoFimex::CDMReader::CDMReader () [inline]

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

11.12.3 Member Function Documentation

11.12.3.1 virtual const CDM& MetNoFimex::CDMReader::getCDM () const [inline, virtual]

Reimplemented in MetNoFimex::FeltCDMReader.

References cdm.

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

data-reading function to be called from the CDMWriter

This function 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

Implemented in MetNoFimex::CDMExtractor, MetNoFimex::CDMTimeInterpolator, and MetNoFimex::NetCDF_CF10_CDMReader.

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

data-reading function to be called from the CDMWriter

The getData function is a convenience 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

11.12.3.4 virtual const boost::shared_ptr<Data> MetNoFimex::CDMReader::getDataSliceFromMemory (const CDMVariable & *variable*, size_t *unLimDimPos* = 0) throw (CDMException) [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

11.12.4 Member Data Documentation

11.12.4.1 CDM MetNoFimex::CDMReader::cdm [protected]

Referenced by MetNoFimex::FeltCDMReader::getCDM(), and getCDM().

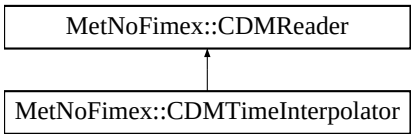
The documentation for this class was generated from the following file:

- include/fimex/CDMReader.h

11.13 MetNoFimex::CDMTimeInterpolator Class Reference

`#include <CDMTimeInterpolator.h>`

Inheritance diagram for MetNoFimex::CDMTimeInterpolator::



Public Member Functions

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

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

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

11.13.1 Constructor & Destructor Documentation

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

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

11.13.2 Member Function Documentation

11.13.2.1 virtual const boost::shared_ptr<Data> MetNoFimex::CDMTimeInterpolator::getDataSlice (const std::string & *varName*, size_t *unLimDimPos* = 0) throw (CDMException) `[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.

11.13.2.2 virtual void MetNoFimex::CDMTimeInterpolator::changeTimeAxis (std::string *timeSpec*) throw (CDMException) `[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

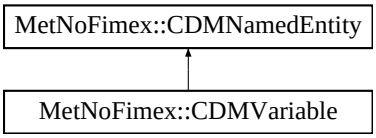
The documentation for this class was generated from the following file:

- include/fimex/CDMTimeInterpolator.h

11.14 MetNoFimex::CDMVariable Class Reference

`#include <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
- CDMDataType getDataType () const
- const **std::vector**< **std::string** > & getShape () const
- 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 data from this variable

- int hasData () const

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

11.14.1 Constructor & Destructor Documentation

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

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

11.14.2 Member Function Documentation

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

Implements MetNoFimex::CDMNamedEntity.

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

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

11.14.2.4 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"

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

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

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

get the spatial counterpart of this vector

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

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

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

check the dimension of a variable

Parameters:

dimension the dimension to check for

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

print a xml representation to the stream without attributes

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

print a xml representation to the stream with attributes

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

add data to the variable

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

retrieve 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

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

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

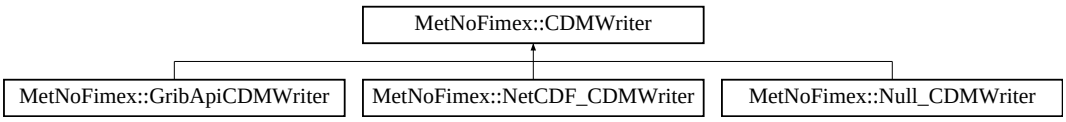
The documentation for this class was generated from the following file:

- include/fimex/CDMVariable.h

11.15 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

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

11.15.1 Constructor & Destructor Documentation

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

11.15.1.2 virtual MetNoFimex::CDMWriter::~CDMWriter () `[inline, virtual]`

11.15.2 Member Data Documentation

11.15.2.1 const boost::shared_ptr<CDMReader> MetNoFimex::CDMWriter::cdmReader `[protected]`

11.15.2.2 const std::string MetNoFimex::CDMWriter::outputFile `[protected]`

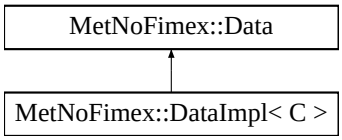
The documentation for this class was generated from the following file:

- include/fimex/CDMWriter.h

11.16 MetNoFimex::Data Class Reference

`#include <Data.h>`

Inheritance diagram for MetNoFimex::Data::



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

  retrieve data as char

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

  retrieve data as char

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

  retrieve data as short

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

  retrieve data as short

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

  retrieve data as int

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

  retrieve data as int

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

  retrieve data as float

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

  retrieve data as float (eventually copy)

- virtual const boost::shared_array< 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 last=-1)=0 throw (CDMException)
- virtual boost::shared_ptr< Data > slice (**std::vector**< size_t > orgDimSize, **std::vector**< size_t > startDims, **std::vector**< size_t > outputDimSize)=0 throw (CDMException)

  get a 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 throw (CDMException)

  convert the datatype from one type,fill,scale,offset to another

- virtual CDMDataType getDataType () const =0

11.16.1 Detailed Description

General class for storing different basic array pointers plus length

11.16.2 Constructor & Destructor Documentation

11.16.2.1 virtual MetNoFimex::Data::~Data () [pure virtual]

11.16.3 Member Function Documentation

11.16.3.1 virtual size_t MetNoFimex::Data::size () const [pure virtual]

size of the data

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.2 virtual int MetNoFimex::Data::bytes_for_one () const [pure virtual]

sizeof the data-impl datatype

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.3 virtual void* MetNoFimex::Data::getDataPtr () [pure virtual]

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.4 virtual void MetNoFimex::Data::toStream (std::ostream &, std::string *separator* = "") const [pure virtual]

printing of the current data to ostream, with optional separator

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.5 virtual const boost::shared_array<char> MetNoFimex::Data::asConstChar () const [pure virtual]

retrieve data as char

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.6 virtual boost::shared_array<char> MetNoFimex::Data::asChar () [pure virtual]

retrieve data as char

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.7 virtual const boost::shared_array<short> MetNoFimex::Data::asConstShort () const [pure virtual]

retrieve data as short

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.8 virtual boost::shared_array<short> MetNoFimex::Data::asShort () [pure virtual]

retrieve data as short

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.9 virtual const boost::shared_array<int> MetNoFimex::Data::asConstInt () const [pure virtual]

retrieve data as int

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.10 virtual boost::shared_array<int> MetNoFimex::Data::asInt () [pure virtual]

retrieve data as int

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.11 virtual const boost::shared_array<float> MetNoFimex::Data::asConstFloat () const [pure virtual]

retrieve data as float

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.12 virtual boost::shared_array<float> MetNoFimex::Data::asFloat () [pure virtual]

retrieve data as float (eventually copy)

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.13 virtual const boost::shared_array<double> MetNoFimex::Data::asConstDouble () const [pure virtual]

retrieve data as double

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.14 virtual boost::shared_array<double> MetNoFimex::Data::asDouble () [pure virtual]

retrieve data as double

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.15 virtual std::string MetNoFimex::Data::asString (std::string *separator* = " ") const [pure virtual]

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

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.16 virtual void MetNoFimex::Data::setValue (long *pos*, double *val*) [pure virtual]

set a value at the desired position

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.17 virtual void MetNoFimex::Data::setValues (size_t *startPos*, const Data & *data*, size_t *first* = 0, size_t *last* = -1) throw (CDMException) [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

last the last (excluded) data-entry, defaults to MAX size_t, automatically shrunken to fit size

Implemented in MetNoFimex::DataImpl< C >, MetNoFimex::DataImpl< C >, MetNoFimex::DataImpl< C >, MetNoFimex::DataImpl< C >, MetNoFimex::DataImpl< C >, and MetNoFimex::DataImpl< C >.

11.16.3.18 virtual boost::shared_ptr<Data> MetNoFimex::Data::slice (std::vector< size_t > *orgDimSize*, std::vector< size_t > *startDims*, std::vector< size_t > *outputDimSize*) throw (CDMException) [pure virtual]

get a 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)

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.19 virtual boost::shared_ptr<Data> MetNoFimex::Data::convertDataType (double *oldFill*, double *oldScale*, double *oldOffset*, CDMDataType *newType*, double *newFill*, double *newScale*, double *newOffset*) throw (CDMException) [pure virtual]

convert the datatype from one type,fill,scale,offset to another

Implemented in MetNoFimex::DataImpl< C >.

11.16.3.20 virtual CDMDataType MetNoFimex::Data::getDataType () const [pure virtual]

return the CDMDataType of this data

Implemented in MetNoFimex::DataImpl< C >.

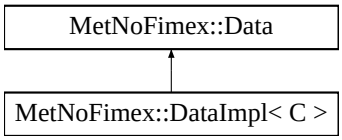
The documentation for this class was generated from the following file:

- include/fimex/Data.h

11.17 MetNoFimex::DataImpl< C > Class Template Reference

`#include <DataImpl.h>`

Inheritance diagram for MetNoFimex::DataImpl< C >::



Public Member Functions

- DataImpl (long length)

  constructor where the array will be automatically allocated

- DataImpl (boost::shared_array< C > array, long length)
- virtual ~DataImpl ()
- virtual size_t size () const

  size of the data

- virtual int bytes_for_one () const

  sizeof the data-impl datatype

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

  printing of the current data to ostream, with optional separator

- virtual const boost::shared_array< C > asBase () const

  get the datapointer of the data

- template<typename T>
  const boost::shared_array< T > as () const
- template<typename T>
  boost::shared_array< T > as ()
- virtual const boost::shared_array< char > asConstChar () const

  retrieve data as char

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

  retrieve data as char

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

  retrieve data as short

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

  retrieve data as short

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

  retrieve data as int

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

  retrieve data as int

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

  retrieve data as float

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

  retrieve data as float (eventually copy)

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

  retrieve data as double

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

  retrieve data as double

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

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

- virtual void setValue (long pos, double val)

  set a value at the desired position

- virtual void setValues (size_t startPos, const Data &data, size_t first=0, size_t last=-1) throw (CDMException)
- virtual boost::shared_ptr< Data > slice (**std::vector**< size_t > orgDimSize, **std::vector**< size_t > startDims, **std::vector**< size_t > outputDimSize) throw (CDMException)

  get a slice of the data

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

  convert the datatype from one type,fill,scale,offset to another

- virtual CDMDataType getDataType () const
- template<class InputIterator>
  void setValues (InputIterator first, InputIterator last, size_t dataStartPos=0) throw (CDMException)
- template<>
  void setValues (size_t startPos, const Data &data, size_t first, size_t last) throw(CDMException)
- template<>
  void setValues (size_t startPos, const Data &data, size_t first, size_t last) throw(CDMException)
- template<>
  void setValues (size_t startPos, const Data &data, size_t first, size_t last) throw(CDMException)
- template<>
  void setValues (size_t startPos, const Data &data, size_t first, size_t last) throw(CDMException)
- template<>
  void setValues (size_t startPos, const Data &data, size_t first, size_t last) throw(CDMException)

template<typename C> class MetNoFimex::DataImpl< C >

11.17.1 Constructor & Destructor Documentation

11.17.1.1 template<typename C> MetNoFimex::DataImpl< C >::DataImpl (long *length*) [inline, explicit]

constructor where the array will be automatically allocated

11.17.1.2 template<typename C> MetNoFimex::DataImpl< C >::DataImpl (boost::shared_array< C > *array*, long *length*) [inline, explicit]

11.17.1.3 template<typename C> virtual MetNoFimex::DataImpl< C >::~DataImpl () [inline, virtual]

11.17.2 Member Function Documentation

11.17.2.1 template<typename C> virtual size_t MetNoFimex::DataImpl< C >::size () const [inline, virtual]

size of the data

Implements MetNoFimex::Data.

11.17.2.2 template<typename C> virtual int MetNoFimex::DataImpl< C >::bytes_for_one () const [inline, virtual]

sizeof the data-impl datatype

Implements MetNoFimex::Data.

11.17.2.3 template<typename C> virtual void* MetNoFimex::DataImpl< C >::getDataPtr () [inline, virtual]

Implements MetNoFimex::Data.

11.17.2.4 template<typename C> void MetNoFimex::DataImpl< C >::toStream (std::ostream &, std::string *separator* = " ") const [inline, virtual]

printing of the current data to ostream, with optional separator

Implements MetNoFimex::Data.

Referenced by MetNoFimex::DataImpl< C >::asString().

11.17.2.5 template<typename C> virtual const boost::shared_array<C> MetNoFimex::DataImpl< C >::asBase () const [inline, virtual]

get the datapointer of the data

11.17.2.6 template<typename C> template<typename T> const boost::shared_array<T> MetNoFimex::DataImpl< C >::as () const [inline]

general conversion function, not in base since template methods not allowed

11.17.2.7 template<typename C> template<typename T> boost::shared_array<T> MetNoFimex::DataImpl< C >::as () [inline]

11.17.2.8 template<typename C> virtual const boost::shared_array<char> MetNoFimex::DataImpl< C >::asConstChar () const [inline, virtual]

retrieve data as char

Implements MetNoFimex::Data.

11.17.2.9 template<typename C> virtual boost::shared_array<char> MetNoFimex::DataImpl< C >::asChar () [inline, virtual]

retrieve data as char

Implements MetNoFimex::Data.

11.17.2.10 template<typename C> virtual const boost::shared_array<short> MetNoFimex::DataImpl< C >::asConstShort () const [inline, virtual]

retrieve data as short

Implements MetNoFimex::Data.

11.17.2.11 template<typename C> virtual boost::shared_array<short> MetNoFimex::DataImpl< C >::asShort () [inline, virtual]

retrieve data as short

Implements MetNoFimex::Data.

11.17.2.12 template<typename C> virtual const boost::shared_array<int> MetNoFimex::DataImpl< C >::asConstInt () const [inline, virtual]

retrieve data as int

Implements MetNoFimex::Data.

11.17.2.13 template<typename C> virtual boost::shared_array<int> MetNoFimex::DataImpl< C >::asInt () [inline, virtual]

retrieve data as int

Implements MetNoFimex::Data.

11.17.2.14 template<typename C> virtual const boost::shared_array<float> MetNoFimex::DataImpl< C >::asConstFloat () const [inline, virtual]

retrieve data as float

Implements MetNoFimex::Data.

11.17.2.15 template<typename C> virtual boost::shared_array<float> MetNoFimex::DataImpl< C >::asFloat () [inline, virtual]

retrieve data as float (eventually copy)

Implements MetNoFimex::Data.

11.17.2.16 template<typename C> virtual const boost::shared_array<double> MetNoFimex::DataImpl< C >::asConstDouble () const [inline, virtual]

retrieve data as double

Implements MetNoFimex::Data.

11.17.2.17 template<typename C> virtual boost::shared_array<double> MetNoFimex::DataImpl< C >::asDouble () [inline, virtual]

retrieve data as double

Implements MetNoFimex::Data.

11.17.2.18 template<typename C> std::string MetNoFimex::DataImpl< C >::asString (std::string *separator* = "") const [inline, virtual]

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

Implements MetNoFimex::Data.

References std::basic_ostringstream< _CharT, _Traits, _Alloc >::str(), and MetNoFimex::DataImpl< C >::toStream().

11.17.2.19 template<typename C> virtual void MetNoFimex::DataImpl< C >::setValue (long *pos*, double *val*) [inline, virtual]

set a value at the desired position

Implements MetNoFimex::Data.

11.17.2.20 template<typename C> void MetNoFimex::DataImpl< C >::setValues (size_t *startPos*, const Data & *data*, size_t *first* = 0, size_t *last* = -1) throw (CDMException) [inline, 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

last the last (excluded) data-entry, defaults to MAX size_t, automatically shrunken to fit size

Implements MetNoFimex::Data.

11.17.2.21 template<typename C> boost::shared_ptr< Data > MetNoFimex::DataImpl< C >::slice (std::vector< size_t > *orgDimSize*, std::vector< size_t > *startDims*, std::vector< size_t > *outputDimSize*) throw (CDMException) [inline, virtual]

get a 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)

Implements MetNoFimex::Data.

References MetNoFimex::recursiveCopyMultiDimData(), std::vector< _Tp, _Alloc >::reserve(), std::vector< _Tp, _Alloc >::size(), and MetNoFimex::type2string().

11.17.2.22 template<typename C> boost::shared_ptr< Data > MetNoFimex::DataImpl< C >::convertDataType (double *oldFill*, double *oldScale*, double *oldOffset*, CDMDataType *newType*, double *newFill*, double *newScale*, double *newOffset*) throw (CDMException) [inline, virtual]

convert the datatype from one type,fill,scale,offset to another

Implements MetNoFimex::Data.

References MetNoFimex::CDM_CHAR, MetNoFimex::CDM_DOUBLE, MetNoFimex::CDM_-FLOAT, MetNoFimex::CDM_INT, MetNoFimex::CDM_NAT, MetNoFimex::CDM_SHORT, and MetNoFimex::CDM_STRING.

11.17.2.23 template<typename C> virtual CDMDataType MetNoFimex::DataImpl< C >::getDataType () const [inline, virtual]

return the CDMDataType of this data

Implements MetNoFimex::Data.

References MetNoFimex::CDM_NAT.

11.17.2.24 template<typename C> template<class InputIterator> void MetNoFimex::DataImpl< C >::setValues (InputIterator *first*, InputIterator *last*, size_t *dataStartPos* = 0) throw (CDMException) [inline]

set the values of the data by the input-iterator

References MetNoFimex::type2string().

11.17.2.25 template<> void MetNoFimex::DataImpl< char >::setValues (size_t *startPos*, const Data & *data*, size_t *first*, size_t *last*) throw(CDMException) [inline, 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

last the last (excluded) data-entry, defaults to MAX size_t, automatically shrunken to fit size

Implements MetNoFimex::Data.

11.17.2.26 template<> void MetNoFimex::DataImpl< short >::setValues (size_t *startPos*, const Data & *data*, size_t *first*, size_t *last*) throw(CDMException) [inline, 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

last the last (excluded) data-entry, defaults to MAX size_t, automatically shrunken to fit size

Implements MetNoFimex::Data.

11.17.2.27 template<> void MetNoFimex::DataImpl< int >::setValues (size_t *startPos*, const Data & *data*, size_t *first*, size_t *last*) throw(CDMException) [inline, 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

last the last (excluded) data-entry, defaults to MAX size_t, automatically shrunken to fit size

Implements MetNoFimex::Data.

11.17.2.28 template<> void MetNoFimex::DataImpl< float >::setValues (size_t *startPos*, const Data & *data*, size_t *first*, size_t *last*) throw(CDMException) [inline, 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

last the last (excluded) data-entry, defaults to MAX size_t, automatically shrunken to fit size

Implements MetNoFimex::Data.

11.17.2.29 template<> void MetNoFimex::DataImpl< double >::setValues (size_t *startPos*, const Data & *data*, size_t *first*, size_t *last*) throw(CDMException) [inline, 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

last the last (excluded) data-entry, defaults to MAX size_t, automatically shrunken to fit size

Implements MetNoFimex::Data.

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

- include/fimex/DataImpl.h

11.18 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

11.18.1 Detailed Description

brief wrapper class around data->convertType

11.18.2 Constructor & Destructor Documentation

11.18.2.1 MetNoFimex::DataTypeChanger::DataTypeChanger (CDMDataType *oldType*) [explicit]

initialize data with the oldType convertData will do nothing in this case

11.18.2.2 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.

11.18.2.3 virtual MetNoFimex::DataTypeChanger::~DataTypeChanger () [virtual]

11.18.3 Member Function Documentation

11.18.3.1 boost::shared_ptr<Data> MetNoFimex::DataTypeChanger::convertData (boost::shared_ptr< Data >) const throw (CDMException)

convert the data to the new scale/fill/offset

11.18.3.2 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

11.19 MetNoFelt::Felt_Array Class Reference

encapsulate parameters of a felt file

```
#include <Felt_Array.h>
```

Public Member Functions

- Felt_Array ()
- Felt_Array (const **string** name, const boost::array< short, 16 > idx, const **string** &dataType="short")
- virtual ~Felt_Array ()
- void addInformationByIndex (const boost::array< short, 16 > &idx, int fieldSize) throw (Felt_File_-Error)
- const boost::array< short, 16 > & getIndexHeader () const

  get the time/level independent index of this header

- void setDataHeader (boost::array< short, 20 > header) throw (Felt_File_Error)

  set the felt data-header for this array a Felt_File_Error will be thrown if the header is different for the different times/layers of this Array

- const boost::array< short, 20 > & getDataHeader () const

  get the time/level independent data-header

- short getLevelType () const

  get the felt level type of this array

- void setGridType (int gridType)

  set the gridType as used in libmi gridPar function

- int getGridType () const

  get the gridType

- void setGridParameters (boost::array< float, 6 > gridParameters)
- const boost::array< float, 6 > & getGridParameters () const

  get the extra grid information from the end of the data

- const **string** & getName () const
- const **string** & getDatatype () const
- double getFillValue () const
- void setFillValue (double fillValue)
- **vector**< time_t > getTimes () const
- **vector**< short > getLevels () const
- **vector**< **pair**< short, short > > getLevelPairs () const
- void addIdent19 (time_t time, **pair**< short, short > levelPair, short value)
- short getIdent19 (time_t time, **pair**< short, short > levelPair) const throw (Felt_File_Error)
- short getIdent19 (**pair**< short, short > levelPair) const throw (Felt_File_Error)
- short getIdent19 (time_t time) const throw (Felt_File_Error)
- short getIdent19 () const throw (Felt_File_Error)
- int getX () const

- int getY () const
- short getVerticalFeltType () const
- double getScalingFactor () const
- boost::array< short, 16 > const getIndex (time_t time, short level) throw (Felt_File_Error)
- int getFieldSize (time_t time, short level) const throw (Felt_File_Error)

11.19.1 Detailed Description

encapsulate parameters of a felt file

store local variables of a parameter, partially retrieved from the diana.setup, partially retrieved from the file

11.19.2 Constructor & Destructor Documentation

11.19.2.1 MetNoFelt::Felt_Array::Felt_Array ()

constructor

11.19.2.2 MetNoFelt::Felt_Array::Felt_Array (const string *name*, const boost::array< short, 16 > *idx*, const string & *dataType* = "short") [explicit]

constructor applying the parameter name and the felt description index array

Parameters:

name parameter name

idx feltfiles are indexed by a short[16] arrays representing different parameters. The parameters used here are those applied to qfelt (query felt)

dataType short|float|double datatype used for autoscaling, getScalingFactor() will be allways return 1 for float and double

11.19.2.3 virtual MetNoFelt::Felt_Array::~Felt_Array () [virtual]

11.19.3 Member Function Documentation

11.19.3.1 void MetNoFelt::Felt_Array::addInformationByIndex (const boost::array< short, 16 > & *idx*, int *fieldSize*) throw (Felt_File_Error)

add information from the felt-index (usually retrieved from qfelt) to this Felt_Array the index given here must correspond to the initialization index

11.19.3.2 const boost::array<short, 16>& MetNoFelt::Felt_Array::getIndexHeader () const [inline]

get the time/level independent index of this header

11.19.3.3 void MetNoFelt::Felt_Array::setDataHeader (boost::array< short, 20 > *header*) throw (Felt_File_Error)

set the felt data-header for this array a Felt_File_Error will be thrown if the header is different for the different times/layers of this Array

Exceptions:

Felt_File_Error when data-definitions change

11.19.3.4 const boost::array<short, 20>& MetNoFelt::Felt_Array::getDataHeader () const [inline]

get the time/level independent data-header

11.19.3.5 short MetNoFelt::Felt_Array::getLevelType () const [inline]

get the felt level type of this array

11.19.3.6 void MetNoFelt::Felt_Array::setGridType (int *gridType*) [inline]

set the gridType as used in libmi gridPar function

11.19.3.7 int MetNoFelt::Felt_Array::getGridType () const [inline]

get the gridType

11.19.3.8 void MetNoFelt::Felt_Array::setGridParameters (boost::array< float, 6 > *gridParameters*) [inline]

set all the grid parameters from the felt file as retrieved from libmi's gridPar function

11.19.3.9 const boost::array<float, 6>& MetNoFelt::Felt_Array::getGridParameters () const [inline]

get the extra grid information from the end of the data

11.19.3.10 const string& MetNoFelt::Felt_Array::getName () const

return the parameter name

11.19.3.11 const string& MetNoFelt::Felt_Array::getDatatype () const [inline]

return the datatype as string short|float|double

11.19.3.12 double MetNoFelt::Felt_Array::getFillValue () const [inline]

return the changed fill used in Felt_File::getScaledDataSlice

11.19.3.13 void MetNoFelt::Felt_Array::setFillValue (double *fillValue*) [inline]

set the fill value to be used in Felt_File::getScaledDataSlice

11.19.3.14 vector<time_t> MetNoFelt::Felt_Array::getTimes () const

return the times available for this parameter, sorted

11.19.3.15 vector<short> MetNoFelt::Felt_Array::getLevels () const

return the levels available for this parameter, sorted

11.19.3.16 vector<pair<short, short> > MetNoFelt::Felt_Array::getLevelPairs () const

return the level pairs (niveau 1, niveau 2) for this parameter as used by hybrid levels

11.19.3.17 void MetNoFelt::Felt_Array::addIdent19 (time_t *time*, pair< short, short > *levelPair*, short *value*) [inline]

add the ident19 parameter from the data-header

11.19.3.18 short MetNoFelt::Felt_Array::getIdent19 (time_t *time*, pair< short, short > *levelPair*) const throw (Felt_File_Error)

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

11.19.3.19 short MetNoFelt::Felt_Array::getIdent19 (pair< short, short > *levelPair*) const throw (Felt_File_Error)

get the ident19 parameter from the data-header, assures that the parameters keep constant across all times for each levelPair or throws a Felt_File_Error

Warning:

only ident19 of data already read will be taken into account

11.19.3.20 short MetNoFelt::Felt_Array::getIdent19 (time_t *time*) const throw (Felt_File_Error)

get the ident19 parameter from the data-header, assures that the parameters keep constant across all levelPair for each time or throws a Felt_File_Error

Warning:

only ident19 of data already read will be taken into account

11.19.3.21 short MetNoFelt::Felt_Array::getIdent19 () const throw (Felt_File_Error)

get the ident19 parameter from the data-header, assures that the parameters keep constant across all level-Pair and times or throws a Felt_File_Error

Warning:

only ident19 of data already read will be taken into account

11.19.3.22 int MetNoFelt::Felt_Array::getX () const [inline]

return x/longitude size

11.19.3.23 int MetNoFelt::Felt_Array::getY () const [inline]

return y/latitude size

11.19.3.24 short MetNoFelt::Felt_Array::getVerticalFeltType () const [inline]

return the felt-type of the vertical axis

11.19.3.25 double MetNoFelt::Felt_Array::getScalingFactor () const

return scalingFactor

11.19.3.26 boost::array<short, 16> const MetNoFelt::Felt_Array::getIndex (time_t *time*, short *level*) throw (Felt_File_Error)

return a copy of the index used within this Felt_Array

11.19.3.27 int MetNoFelt::Felt_Array::getFieldSize (time_t *time*, short *level*) const throw (Felt_File_Error)

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

- include/fimex/Felt_Array.h

11.20 MetNoFelt::Felt_File Class Reference

Felt File access.

```
#include <Felt_File.h>
```

Public Member Functions

- Felt_File ()

  constructor

- Felt_File (const **std::string** &filename) throw (Felt_File_Error)
- Felt_File (const **std::string** &filename, const **std::vector**< **std::string** > &dianaParamList, const **std::map**< **std::string**, **std::string** > &options) throw (Felt_File_Error)
- virtual ~Felt_File ()
- Felt_Array & getFeltArray (const **std::string** &compName) throw (Felt_File_Error)

  retrieve a Felt_Array

- **std::vector**< short > getDataSlice (const **std::string** &compName, const std::time_t time, const short level) throw (Felt_File_Error)

  retrieve a data slice

- boost::shared_ptr< MetNoFimex::Data > getScaledDataSlice (const **std::string** &compName, const std::time_t time, const short level, double fillValue) throw (Felt_File_Error)
- **std::vector**< Felt_Array > listFeltArrays ()
- **std::map**< short, **std::vector**< short > > getFeltLevels () const
- **std::map**< short, **std::vector**< **pair**< short, short > > > getFeltLevelPairs () const
- const **ShortPairMap** & getHybridLevels () const
- **std::vector**< time_t > getFeltTimes () const

  all time values, sorted

- int getNX () const

  get size in x direction

- int getNY () const

  get size in y direction

- boost::shared_ptr< MetNoFimex::Data > getXData () const throw (Felt_File_Error)

  get the values of the x axis

- boost::shared_ptr< MetNoFimex::Data > getYData () const throw (Felt_File_Error)

  get the values of the y axis

- short getGridType () const throw (Felt_File_Error)

  assumes one set of grid-type for the whole file

- const boost::array< float, 6 > & getGridParameters () const throw (Felt_File_Error)

  assumes one set of grid-parameters for the whole file

11.20.1 Detailed Description

Felt File access.

Felt_File gives c++ style access to felt files. It uses internally libmi and caches the table of contents

11.20.2 Constructor & Destructor Documentation

11.20.2.1 MetNoFelt::Felt_File::Felt_File () [inline]

constructor

open an empty felt file, just a default constructor, no useful information

11.20.2.2 MetNoFelt::Felt_File::Felt_File (const std::string & *filename*) throw (Felt_File_Error) [explicit]

open and read toc of a felt file

Parameters:

filename name of felt file

11.20.2.3 MetNoFelt::Felt_File::Felt_File (const std::string & *filename*, const std::vector< std::string > & *dianaParamList*, const std::map< std::string, std::string > & *options*) throw (Felt_File_Error) [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

11.20.2.4 virtual MetNoFelt::Felt_File::~Felt_File () [virtual]

11.20.3 Member Function Documentation

11.20.3.1 Felt_Array& MetNoFelt::Felt_File::getFeltArray (const std::string & *compName*) throw (Felt_File_Error)

retrieve a Felt_Array

Parameters:

compName parameter name of felt file as named in diana setup

11.20.3.2 std::vector<short> MetNoFelt::Felt_File::getDataSlice (const std::string & *compName*, const std::time_t *time*, const short *level*) throw (Felt_File_Error)

retrieve a data slice

Parameters:

compName parameter name of felt file

time time of slice

level level of slice

11.20.3.3 boost::shared_ptr<MetNoFimex::Data> MetNoFelt::Felt_File::getScaledDataSlice (const std::string & *compName*, const std::time_t *time*, const short *level*, double *fillValue*) throw (Felt_File_Error)

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

11.20.3.4 std::vector<Felt_Array> MetNoFelt::Felt_File::listFeltArrays ()

retrieve all felt arrays

11.20.3.5 std::map<short, std::vector<short> > MetNoFelt::Felt_File::getFeltLevels () const

Z-axis types and values

Returns:

map consisting of felt level-ids and a sorted vector of level values

11.20.3.6 std::map<short, std::vector<pair<short,short> > > MetNoFelt::Felt_File::getFeltLevelPairs () const

Z-axis types and values

Returns:

map consisting of felt level-ids and a sorted vector of level-pairs of values

11.20.3.7 const ShortPairMap& MetNoFelt::Felt_File::getHybridLevels () const [inline]

11.20.3.8 std::vector<time_t> MetNoFelt::Felt_File::getFeltTimes () const

all time values, sorted

11.20.3.9 int MetNoFelt::Felt_File::getNX () const

get size in x direction

11.20.3.10 int MetNoFelt::Felt_File::getNY () const

get size in y direction

11.20.3.11 boost::shared_ptr<MetNoFimex::Data> MetNoFelt::Felt_File::getXData () const throw (Felt_File_Error)

get the values of the x axis

11.20.3.12 boost::shared_ptr<MetNoFimex::Data> MetNoFelt::Felt_File::getYData () const throw (Felt_File_Error)

get the values of the y axis

11.20.3.13 short MetNoFelt::Felt_File::getGridType () const throw (Felt_File_Error)

assumes one set of grid-type for the whole file

11.20.3.14 const boost::array<float, 6>& MetNoFelt::Felt_File::getGridParameters () const throw (Felt_File_Error)

assumes one set of grid-parameters for the whole file

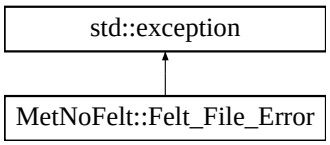
The documentation for this class was generated from the following file:

- include/fimex/Felt_File.h

11.21 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)
- virtual ~Felt_File_Error () throw ()
- virtual const char * what () const throw ()

11.21.1 Constructor & Destructor Documentation

11.21.1.1 MetNoFelt::Felt_File_Error::Felt_File_Error (const std::string & *message*) `[explicit]`

11.21.1.2 virtual MetNoFelt::Felt_File_Error::~Felt_File_Error () throw () `[virtual]`

11.21.2 Member Function Documentation

11.21.2.1 virtual const char* MetNoFelt::Felt_File_Error::what () const throw () `[virtual]`

Reimplemented from **std::exception**.

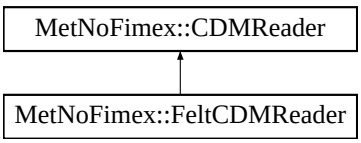
The documentation for this class was generated from the following file:

- include/fimex/Felt_File_Error.h

11.22 MetNoFimex::FeltCDMReader Class Reference

`#include <FeltCDMReader.h>`

Inheritance diagram for MetNoFimex::FeltCDMReader::



Public Member Functions

- FeltCDMReader (**std::string** filename, **std::string** configFilename) throw (CDMException)
- virtual ~FeltCDMReader ()
- virtual const boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos) throw (CDMException)
- virtual const CDM & getCDM () const

11.22.1 Constructor & Destructor Documentation

11.22.1.1 MetNoFimex::FeltCDMReader::FeltCDMReader (std::string *filename*, std::string *configFilename*) throw (CDMException)

11.22.1.2 virtual MetNoFimex::FeltCDMReader::~FeltCDMReader () `[virtual]`

11.22.2 Member Function Documentation

11.22.2.1 virtual const boost::shared_ptr<Data> MetNoFimex::FeltCDMReader::getDataSlice (const std::string & *varName*, size_t *unLimDimPos*) throw (CDMException) `[virtual]`

11.22.2.2 virtual const CDM& MetNoFimex::FeltCDMReader::getCDM () const `[inline, virtual]`

Reimplemented from MetNoFimex::CDMReader.

References MetNoFimex::CDMReader::cdm.

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

- include/fimex/FeltCDMReader.h

11.23 MetNoFelt::FeltParameters Class Reference

```
#include <FeltParameters.h>
```

Public Member Functions

- FeltParameters ()
- FeltParameters (**std::string** filename)
- FeltParameters (const **std::vector**< **std::string** > &feltParams)
- 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 ()

11.23.1 Constructor & Destructor Documentation

11.23.1.1 MetNoFelt::FeltParameters::FeltParameters ()

11.23.1.2 MetNoFelt::FeltParameters::FeltParameters (std::string *filename*) [explicit]

initialize all known felt parameters from a diana-setup file

Parameters:

filename diana setup file

11.23.1.3 MetNoFelt::FeltParameters::FeltParameters (const std::vector< std::string > & *feltParams*) [explicit]

initialize parameters from a list of parameters in diana format, e.g. 17,2,1000:prod=74

11.23.1.4 virtual MetNoFelt::FeltParameters::~FeltParameters () [virtual]

11.23.2 Member Function Documentation

11.23.2.1 const boost::array<short, 16>& MetNoFelt::FeltParameters::getParameters (const std::string &)

11.23.2.2 const std::string& MetNoFelt::FeltParameters::getParameterName (const boost::array< short, 16 > &)

11.23.2.3 std::string MetNoFelt::FeltParameters::getParameterDatatype (const std::string & *parameterName*) const

11.23.2.4 double MetNoFelt::FeltParameters::getParameterFillValue (const std::string & *parameterName*) const

11.23.2.5 static const std::string& MetNoFelt::FeltParameters::DEFAULT_CONFIG () [inline, static]

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

- include/fimex/FeltParameters.h

11.24 MetNoFimex::FimexTime Class Reference

```
#include <TimeUnit.h>
```

Public Member Functions

- 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

Public Attributes

- unsigned short msecond

  millisecond

- char second

  second (0-59)

- char minute

  minute (0-59)

- char hour

  hour (0-23)

- char mday

  day of month (1-31)

- char month

  month (1-12)

- unsigned short year

  year (2008 as of writing)

11.24.1 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

11.24.2 Member Function Documentation

11.24.2.1 bool MetNoFimex::FimexTime::operator== (const FimexTime & *rhs*) const

compare two fimexTimes

11.24.2.2 bool MetNoFimex::FimexTime::operator!= (const FimexTime & *rhs*) const [inline]

compare two fimexTimes

11.24.2.3 bool MetNoFimex::FimexTime::operator> (const FimexTime & *rhs*) const [inline]

compare two fimexTimes

References toLong().

11.24.2.4 bool MetNoFimex::FimexTime::operator< (const FimexTime & *rhs*) const [inline]

compare two fimexTimes

11.24.2.5 bool MetNoFimex::FimexTime::operator>= (const FimexTime & *rhs*) const [inline]

compare two fimexTimes

11.24.2.6 bool MetNoFimex::FimexTime::operator<= (const FimexTime & *rhs*) const [inline]

compare two fimexTimes

11.24.3 Member Data Documentation

11.24.3.1 unsigned short MetNoFimex::FimexTime::msecond

millisecond

11.24.3.2 char MetNoFimex::FimexTime::second

second (0-59)

11.24.3.3 char MetNoFimex::FimexTime::minute

minute (0-59)

11.24.3.4 char MetNoFimex::FimexTime::hour

hour (0-23)

11.24.3.5 char MetNoFimex::FimexTime::mday

day of month (1-31)

11.24.3.6 char MetNoFimex::FimexTime::month

month (1-12)

11.24.3.7 unsigned short MetNoFimex::FimexTime::year

year (2008 as of writing)

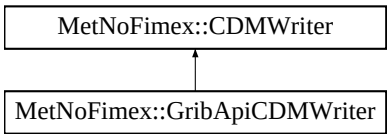
The documentation for this class was generated from the following file:

- include/fimex/TimeUnit.h

11.25 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 ()

11.25.1 Constructor & Destructor Documentation

11.25.1.1 MetNoFimex::GribApiCDMWriter::GribApiCDMWriter (const boost::shared_ptr< CDMReader > *cdmReader*, const std::string & *outputFile*, const int *gribVersion*, const std::string & *configFile*)

11.25.1.2 virtual MetNoFimex::GribApiCDMWriter::~GribApiCDMWriter () `[virtual]`

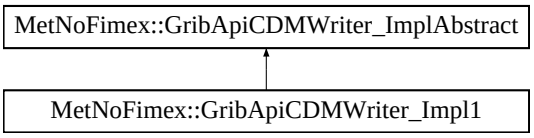
The documentation for this class was generated from the following file:

- include/fimex/GribApiCDMWriter.h

11.26 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)

11.26.1 Detailed Description

Implemenationn of a writer using GribApi for grib1

11.26.2 Constructor & Destructor Documentation

11.26.2.1 MetNoFimex::GribApiCDMWriter_Impl1::GribApiCDMWriter_Impl1 (const boost::shared_ptr< CDMReader > & *cdmReader*, const std::string & *outputFile*, const std::string & *configFile*)

11.26.2.2 virtual MetNoFimex::GribApiCDMWriter_Impl1::~GribApiCDMWriter_Impl1 () `[virtual]`

11.26.3 Member Function Documentation

11.26.3.1 virtual void MetNoFimex::GribApiCDMWriter_Impl1::setParameter (const std::string & *varName*, const FimexTime & *fTime*, double *levelValue*) throw (CDMException) `[virtual]`

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

11.26.3.2 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.

11.26.3.3 virtual void MetNoFimex::GribApiCDMWriter_Impl1::setLevel (const std::string & *varName*, double *levelValue*) [virtual]

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

11.26.3.4 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.

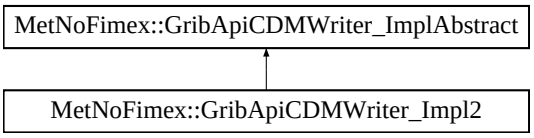
The documentation for this class was generated from the following file:

- include/fimex/GribApiCDMWriter_Impl1.h

11.27 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)

11.27.1 Detailed Description

Implemenationn of a writer using GribApi for grib2

11.27.2 Constructor & Destructor Documentation

11.27.2.1 MetNoFimex::GribApiCDMWriter_Impl2::GribApiCDMWriter_Impl2 (const boost::shared_ptr< CDMReader > & *cdmReader*, const std::string & *outputFile*, const std::string & *configFile*)

11.27.2.2 virtual MetNoFimex::GribApiCDMWriter_Impl2::~GribApiCDMWriter_Impl2 () `[virtual]`

11.27.3 Member Function Documentation

11.27.3.1 virtual void MetNoFimex::GribApiCDMWriter_Impl2::setParameter (const std::string & *varName*, const FimexTime & *fTime*, double *levelValue*) throw (CDMException) `[virtual]`

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

11.27.3.2 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.

11.27.3.3 virtual void MetNoFimex::GribApiCDMWriter_Impl2::setLevel (const std::string & *varName*, double *levelValue*) [virtual]

Implements MetNoFimex::GribApiCDMWriter_ImplAbstract.

11.27.3.4 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.

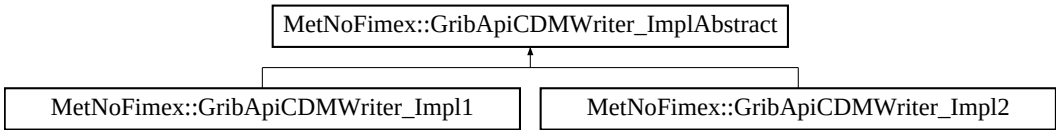
The documentation for this class was generated from the following file:

- include/fimex/GribApiCDMWriter_Impl2.h

11.28 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 boost::shared_ptr< CDMReader > cdmReader
- const **std::string** outputFile
- const **std::string** configFile
- const boost::shared_ptr< XMLDoc > xmlConfig
- boost::shared_ptr< grib_handle > gribHandle
- LoggerPtr logger

11.28.1 Constructor & Destructor Documentation

11.28.1.1 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

11.28.1.2 virtual MetNoFimex::GribApiCDMWriter_ImplAbstract::~GribApiCDMWriter_ImplAbstract () [virtual]

11.28.2 Member Function Documentation

11.28.2.1 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.

11.28.2.2 virtual void MetNoFimex::GribApiCDMWriter_ImplAbstract::setGlobalAttributes () [protected, virtual]

add the global attributes from the config to the default grib-handle

11.28.2.3 virtual void MetNoFimex::GribApiCDMWriter_ImplAbstract::setData (const boost::shared_ptr< Data > & *data*) [protected, virtual]

11.28.2.4 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.

11.28.2.5 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.

11.28.2.6 virtual void MetNoFimex::GribApiCDMWriter_ImplAbstract::setTime (const std::string & *varName*, const FimexTime & *fTime*) [protected, virtual]

11.28.2.7 virtual void MetNoFimex::GribApiCDMWriter_ImplAbstract::setLevel (const std::string & *varName*, double *levelValue*) [protected, pure virtual]

Implemented in MetNoFimex::GribApiCDMWriter_Impl1, and MetNoFimex::GribApiCDMWriter_-Impl2.

11.28.2.8 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

11.28.2.9 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

11.28.2.10 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.

11.28.2.11 virtual void MetNoFimex::GribApiCDMWriter_-ImplAbstract::writeGribHandleToFile () [protected, virtual]

11.28.2.12 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

11.28.3 Member Data Documentation

11.28.3.1 int MetNoFimex::GribApiCDMWriter_ImplAbstract::gribVersion [protected]

11.28.3.2 const boost::shared_ptr<CDMReader> MetNoFimex::GribApiCDMWriter_-ImplAbstract::cdmReader [protected]

11.28.3.3 const std::string MetNoFimex::GribApiCDMWriter_ImplAbstract::outputFile [protected]

11.28.3.4 const std::string MetNoFimex::GribApiCDMWriter_ImplAbstract::configFile [protected]

11.28.3.5 const boost::shared_ptr<XMLDoc> MetNoFimex::GribApiCDMWriter_-ImplAbstract::xmlConfig [protected]

11.28.3.6 boost::shared_ptr<grib_handle> MetNoFimex::GribApiCDMWriter_-ImplAbstract::gribHandle [protected]

11.28.3.7 LoggerPtr MetNoFimex::GribApiCDMWriter_ImplAbstract::logger [protected]

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

- include/fimex/GribApiCDMWriter_ImplAbstract.h

11.29 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)

11.29.1 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.

11.29.2 Member Enumeration Documentation

11.29.2.1 enum MetNoFimex::Logger::LogLevel

different log levels

Enumerator:

OFF
FATAL
ERROR
WARN
INFO
DEBUG

11.29.3 Constructor & Destructor Documentation

11.29.3.1 MetNoFimex::Logger::Logger (const std::string & *className*)

11.29.3.2 virtual MetNoFimex::Logger::~Logger () `[virtual]`

11.29.4 Member Function Documentation

11.29.4.1 virtual bool MetNoFimex::Logger::isEnabledFor (LogLevel *level*) `[virtual]`

check if the loglevel of this logger is active

11.29.4.2 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__

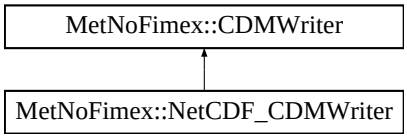
The documentation for this class was generated from the following file:

- include/fimex/Logger.h

11.30 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)
- NetCDF_CDMWriter (const boost::shared_ptr< CDMReader > cdmReader, const **std::string** &outputFile, const **std::string** &configFile)
- virtual ~NetCDF_CDMWriter ()
- const **std::string** & getVariableName (const **std::string** &varName) const
- const **std::string** & getDimensionName (const **std::string** &dimName) const
- const **std::string** & getAttributeName (const **std::string** &varName, const **std::string** &attName) const
- const CDMAttribute & getAttribute (const **std::string** &varName, const **std::string** &attName) const throw (CDMException)

11.30.1 Constructor & Destructor Documentation

11.30.1.1 MetNoFimex::NetCDF_CDMWriter::NetCDF_CDMWriter (const boost::shared_ptr< CDMReader > *cdmReader*, const std::string & *outputFile*)

11.30.1.2 MetNoFimex::NetCDF_CDMWriter::NetCDF_CDMWriter (const boost::shared_ptr< CDMReader > *cdmReader*, const std::string & *outputFile*, const std::string & *configFile*)

Parameters:

cdmReader dataSource

outputFile file-name to write to

configFile xml-configuration

11.30.1.3 virtual MetNoFimex::NetCDF_CDMWriter::~NetCDF_CDMWriter () `[virtual]`

11.30.2 Member Function Documentation

11.30.2.1 const std::string& MetNoFimex::NetCDF_CDMWriter::getVariableName (const std::string & *varName*) const

Returns:

the new name of a variable, eventually changed by the writers config

11.30.2.2 const std::string& MetNoFimex::NetCDF_CDMWriter::getDimensionName (const std::string & *dimName*) const

Returns:

the new name of a dimension, eventually changed by the writers config

11.30.2.3 const std::string& MetNoFimex::NetCDF_CDMWriter::getAttributeName (const std::string & *varName*, const std::string & *attName*) const

Returns:

the new name of an attribute, eventually changed by the writers config

11.30.2.4 const CDMAttribute& MetNoFimex::NetCDF_CDMWriter::getAttribute (const std::string & *varName*, const std::string & *attName*) const throw (CDMException)

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

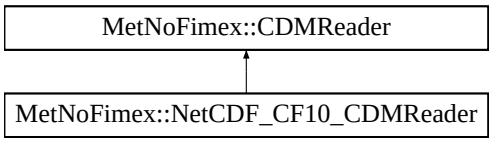
The documentation for this class was generated from the following file:

- include/fimex/NetCDF_CDMWriter.h

11.31 MetNoFimex::NetCDF_CF10_CDMReader Class Reference

`#include <NetCDF_CF10_CDMReader.h>`

Inheritance diagram for MetNoFimex::NetCDF_CF10_CDMReader::



Public Member Functions

- NetCDF_CF10_CDMReader (const **std::string** &fileName)
- virtual ~NetCDF_CF10_CDMReader ()
- virtual const boost::shared_ptr< Data > getDataSlice (const **std::string** &varName, size_t unLimDimPos) throw (CDMException)

  data-reading function to be called from the CDMWriter

11.31.1 Constructor & Destructor Documentation

11.31.1.1 MetNoFimex::NetCDF_CF10_CDMReader::NetCDF_CF10_CDMReader (const std::string & *fileName*)

11.31.1.2 virtual MetNoFimex::NetCDF_CF10_CDMReader::~NetCDF_CF10_CDMReader () `[virtual]`

11.31.2 Member Function Documentation

11.31.2.1 virtual const boost::shared_ptr<Data> MetNoFimex::NetCDF_CF10_CDMReader::getDataSlice (const std::string & *varName*, size_t *unLimDimPos*) throw (CDMException) `[virtual]`

data-reading function to be called from the CDMWriter

This function 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

Implements MetNoFimex::CDMReader.

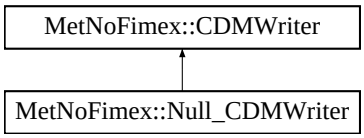
The documentation for this class was generated from the following file:

- include/fimex/NetCDF_CF10_CDMReader.h

11.32 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 ()

11.32.1 Detailed Description

CDMWriter does all operations as the NetCDF_CDMWriter, except writing to the file. This class is useful for performance tests.

11.32.2 Constructor & Destructor Documentation

11.32.2.1 MetNoFimex::Null_CDMWriter::Null_CDMWriter (const boost::shared_ptr< CDMReader > *cdmReader*, const std::string & *outputFile*)

11.32.2.2 virtual MetNoFimex::Null_CDMWriter::~Null_CDMWriter () `[virtual]`

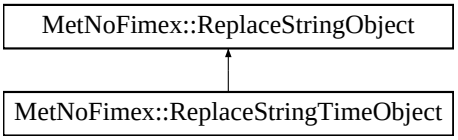
The documentation for this class was generated from the following file:

- include/fimex/Null_CDMWriter.h

11.33 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

11.33.1 Detailed Description

Interface for objects which might be converted to different strings

11.33.2 Constructor & Destructor Documentation

11.33.2.1 virtual MetNoFimex::ReplaceStringObject::~ReplaceStringObject () `[pure virtual]`

11.33.3 Member Function Documentation

11.33.3.1 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.

11.33.3.2 virtual void MetNoFimex::ReplaceStringObject::setFormatString (const std::string & *format*) `[pure virtual]`

set the formatting string for this object

Implemented in MetNoFimex::ReplaceStringTimeObject.

11.33.3.3 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.

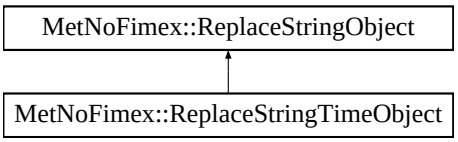
The documentation for this class was generated from the following file:

- include/fimex/ReplaceStringObject.h

11.34 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)

11.34.1 Constructor & Destructor Documentation

11.34.1.1 MetNoFimex::ReplaceStringTimeObject::ReplaceStringTimeObject () `[inline]`

11.34.1.2 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

11.34.1.3 virtual MetNoFimex::ReplaceStringTimeObject::~ReplaceStringTimeObject () `[inline, virtual]`

11.34.2 Member Function Documentation

11.34.2.1 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.

11.34.2.2 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.

11.34.2.3 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.

11.34.3 Friends And Related Function Documentation

11.34.3.1 std::ostream& operator<< (std::ostream & *s*, const ReplaceStringTimeObject & *rsto*) [friend]

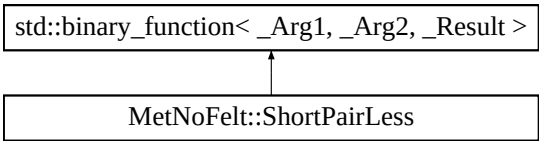
The documentation for this class was generated from the following file:

- include/fimex/ReplaceStringTimeObject.h

11.35 MetNoFelt::ShortPairLess Struct Reference

`#include <Felt_Array.h>`

Inheritance diagram for MetNoFelt::ShortPairLess::



Public Member Functions

- bool operator() (const **pair**< short, short > &p1, const **pair**< short, short > &p2) const

11.35.1 Detailed Description

comparison operator for pair<short, short> used for levelPairs

11.35.2 Member Function Documentation

11.35.2.1 bool MetNoFelt::ShortPairLess::operator() (const pair< short, short > & *p1*, const pair< short, short > & *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_Array.h

11.36 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)

11.36.1 Detailed Description

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

11.36.2 Constructor & Destructor Documentation

11.36.2.1 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

11.36.2.2 virtual MetNoFimex::TimeLevelDataSliceFetcher::~TimeLevelDataSliceFetcher () [virtual]

11.36.3 Member Function Documentation

11.36.3.1 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

11.37 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

11.37.1 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=hours since 2000-01-01 00:0
```

11.37.2 Constructor & Destructor Documentation

11.37.2.1 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

11.37.2.2 virtual MetNoFimex::TimeSpec::~TimeSpec () [inline, virtual]

11.37.3 Member Function Documentation

11.37.3.1 const std::vector<FimexTime>& MetNoFimex::TimeSpec::getTimeSteps () const [inline]

11.37.3.2 const std::string& MetNoFimex::TimeSpec::getUnitString () const [inline]

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

- include/fimex/TimeSpec.h

11.38 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

- 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()***)*

11.38.1 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 FimexTime

All times are assumed to be UTC, and we use the Gregorian Calendar (not 100% true for times before 1600AD, depending on implementation)

11.38.2 Constructor & Destructor Documentation

11.38.2.1 MetNoFimex::TimeUnit::TimeUnit () throw (CDMException)

initialize a timeUnit with a unit string

11.38.2.2 MetNoFimex::TimeUnit::TimeUnit (const std::string & *timeUnitString*) throw (CDMException)

11.38.2.3 virtual MetNoFimex::TimeUnit::~TimeUnit () [virtual]

11.38.3 Member Function Documentation

11.38.3.1 double MetNoFimex::TimeUnit::unitTime2epochSeconds (double *unitTime*) const

calculate the epochSeconds for a time in the current unit

11.38.3.2 double MetNoFimex::TimeUnit::epochSeconds2unitTime (double *epochSeconds*) const

calculate the time in the current unit from the epoch

11.38.3.3 FimexTime MetNoFimex::TimeUnit::unitTime2fimexTime (double *unitTime*) const throw (CDMException)

calculate the time in a calendar form

11.38.3.4 double MetNoFimex::TimeUnit::fimexTime2unitTime (const FimexTime & *fiTime*) const throw (CDMException)

calculate the time in the current unit from the calendar form

Referenced by fimexTime2unitTimeX().

11.38.3.5 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().

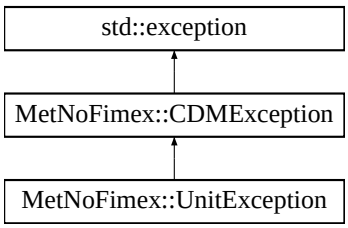
The documentation for this class was generated from the following file:

- include/fimex/TimeUnit.h

11.39 MetNoFimex::UnitException Class Reference

`#include <Units.h>`

Inheritance diagram for MetNoFimex::UnitException::



Public Member Functions

- UnitException ()
- UnitException (**std::string** message)

11.39.1 Constructor & Destructor Documentation

11.39.1.1 MetNoFimex::UnitException::UnitException () `[inline]`

11.39.1.2 MetNoFimex::UnitException::UnitException (std::string *message*) `[inline]`

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

- include/fimex/Units.h

11.40 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 throw (UnitException)

  test if two units are convertible to each others

- bool isTime (const **std::string** &timeUnit) const throw (UnitException)

11.40.1 Constructor & Destructor Documentation

11.40.1.1 MetNoFimex::Units::Units ()

initialization of unit handling, i.e. parsing of unit file etc if required

11.40.1.2 MetNoFimex::Units::Units (const Units & *rhs*)

11.40.1.3 virtual MetNoFimex::Units::~Units () [virtual]

11.40.2 Member Function Documentation

11.40.2.1 Units& MetNoFimex::Units::operator= (const Units & *rhs*)

11.40.2.2 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

11.40.2.3 bool MetNoFimex::Units::areConvertible (const std::string & *unit1*, const std::string & *unit2*) const throw (UnitException)

test if two units are convertible to each others

Parameters:

unit1 first unit

unit2 second unit

11.40.2.4 bool MetNoFimex::Units::isTime (const std::string & *timeUnit*) const throw (UnitException)

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

- include/fimex/Units.h

11.41 MetNoFimex::XMLDoc Class Reference

```
#include <XMLDoc.h>
```

Public Member Functions

- XMLDoc (const **std::string** &filename) throw (CDMException)
- virtual ~XMLDoc ()
- XPathObjPtr getXPathObject (const **std::string** &xpath) const throw (CDMException)

11.41.1 Detailed Description

a tiny wrapper around libxml dom and xpath reader with xml::include

11.41.2 Constructor & Destructor Documentation

11.41.2.1 MetNoFimex::XMLDoc::XMLDoc (const std::string & *filename*) throw (CDMException)

initialization of libxml and the xml config file

Parameters:

filename xml input-file

Exceptions:

CDMException if problems with libxml or problems with input-file

11.41.2.2 virtual MetNoFimex::XMLDoc::~XMLDoc () `[virtual]`

11.41.3 Member Function Documentation

11.41.3.1 XPathObjPtr MetNoFimex::XMLDoc::getXPathObject (const std::string & *xpath*) const throw (CDMException)

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

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

- include/fimex/XMLDoc.h

Chapter 12

File Documentation

12.1 doxydoc.txt File Reference

12.2 include/fimex/CachedInterpolation.h File Reference

```
#include <boost/shared_ptr.hpp>
#include <boost/shared_array.hpp>
#include "fimex/interpolation.h"
#include "fimex/Data.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CachedInterpolation

12.3 include/fimex/CachedVectorReprojection.h File Reference

```
#include <boost/shared_ptr.hpp>
#include "fimex/Data.h"
#include "fimex/interpolation.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CachedVectorReprojection

12.4 include/fimex/CDM.h File Reference

```
#include <map>
#include <vector>
#include <string>
#include <ostream>
#include <boost/regex.hpp>
#include "fimex/CDMAttribute.h"
#include "fimex/CDMVariable.h"
#include "fimex/CDMDimension.h"
#include "fimex/CDMException.h"
#include "fimex/CDMconstants.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CDM

  Data structure of the Common Data Model.

12.5 include/fimex/CDMAttribute.h File Reference

```
#include <string>
#include <ostream>
#include <boost/shared_ptr.hpp>
#include "fimex/Data.h"
#include "fimex/CDMDataType.h"
#include "fimex/CDMNamedEntity.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CDMAttribute

Functions

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

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

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

  convert attributes of a projection-variable to a projString

12.6 include/fimex/CDMconstants.h File Reference

Defines

- #define MIFI_EARTH_RADIUS_M 6371000

12.6.1 Define Documentation

12.6.1.1 #define MIFI_EARTH_RADIUS_M 6371000

12.7 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 }

Functions

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

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

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

12.8 include/fimex/CDMDimension.h File Reference

```
#include <string>
#include <ostream>
#include "fimex/CDMNamedEntity.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CDMDimension

12.9 include/fimex/CDMException.h File Reference

```
#include <exception>
#include <string>
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CDMException

12.10 include/fimex/CDMExtractor.h File Reference

```
#include <boost/array.hpp>
#include <boost/shared_ptr.hpp>
#include "fimex/CDMReader.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CDMExtractor

12.11 include/fimex/CDMInterpolator.h File Reference

```
#include "fimex/CDMReader.h"
#include "fimex/CachedInterpolation.h"
#include "fimex/CachedVectorReprojection.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CDMInterpolator

12.12 include/fimex/CDMNamedEntity.h File Reference

```
#include <string>
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CDMNamedEntity
- struct MetNoFimex::CDMNameCompare
- class MetNoFimex::CDMNameEqual

12.13 include/fimex/CDMReader.h File Reference

```
#include <boost/shared_ptr.hpp>
#include "fimex/CDM.h"
#include "fimex/Data.h"
#include "fimex/CDMVariable.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CDMReader

  Basic interface for CDM reading and manipulation classes.

12.14 include/fimex/CDMTimeInterpolator.h File Reference

```
#include "CDMReader.h"
#include <map>
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CDMTimeInterpolator

12.15 include/fimex/CDMVariable.h File Reference

```
#include <string>
#include <vector>
#include <ostream>
#include "fimex/CDMAttribute.h"
#include "fimex/CDMDimension.h"
#include "fimex/CDMDataType.h"
#include "fimex/CDMNamedEntity.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CDMVariable

12.16 include/fimex/CDMWriter.h File Reference

```
#include <string>
#include <boost/shared_ptr.hpp>
#include "fimex/CDMReader.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::CDMWriter

12.17 include/fimex/config.h File Reference

Defines

- #define HAVE_BOOST
- #define HAVE_BOOST_PROGRAM_OPTIONS
- #define HAVE_BOOST_REGEX
- #define HAVE_BOOST_UNIT_TEST_FRAMEWORK
- #define HAVE_CEIL 1
- #define HAVE_DLFCN_H 1
- #define HAVE_FMOD 1
- #define HAVE_GRIBAPI_H 1
- #define HAVE_INTTYPES_H 1
- #define HAVE_LIBM 1
- #define HAVE_LIBMIC 1
- #define HAVE_LOG10 1
- #define HAVE_MEMORY_H 1
- #define HAVE_MEMSET 1
- #define HAVE_MILIB_MILIB_H 1
- #define HAVE_NAMESPACES
- #define HAVE_NETCDF 1
- #define HAVE_POW 1
- #define HAVE_PROJ4 1
- #define HAVE_SQRT 1
- #define HAVE_STD
- #define HAVE_STDBOOL_H 1
- #define HAVE_STDINT_H 1
- #define HAVE_STDLIB_H 1
- #define HAVE_STL
- #define HAVE_STRINGS_H 1
- #define HAVE_STRING_H 1
- #define HAVE_STRSTR 1
- #define HAVE_SYS_STAT_H 1
- #define HAVE_SYS_TYPES_H 1
- #define HAVE_UDUNITS 1
- #define HAVE_UNISTD_H 1
- #define LSTAT_FOLLOWS_SLASHED_SYMLINK 1
- #define NETCDF_CPP_INCLUDE "/usr/include/netcdfcpp.h"
- #define NETCDF_C_INCLUDE "/usr/include/netcdf.h"
- #define PACKAGE "fimex"
- #define PACKAGE_BUGREPORT "heiko.klein@met.no"
- #define PACKAGE_NAME "fimex"
- #define PACKAGE_STRING "fimex 0.10"
- #define PACKAGE_TARNAME "fimex"
- #define PACKAGE_VERSION "0.10"
- #define STDC_HEADERS 1
- #define VERSION "0.10"

12.17.1 Define Documentation

12.17.1.1 #define HAVE_BOOST

12.17.1.2 #define HAVE_BOOST_PROGRAM_OPTIONS

12.17.1.3 #define HAVE_BOOST_REGEX

12.17.1.4 #define HAVE_BOOST_UNIT_TEST_FRAMEWORK

12.17.1.5 #define HAVE_CEIL 1

12.17.1.6 #define HAVE_DLFCN_H 1

12.17.1.7 #define HAVE_FMOD 1

12.17.1.8 #define HAVE_GRIBAPI_H 1

12.17.1.9 #define HAVE_INTTYPES_H 1

12.17.1.10 #define HAVE_LIBM 1

12.17.1.11 #define HAVE_LIBMIC 1

12.17.1.12 #define HAVE_LOG10 1

12.17.1.13 #define HAVE_MEMORY_H 1

12.17.1.14 #define HAVE_MEMSET 1

12.17.1.15 #define HAVE_MILIB_MILIB_H 1

12.17.1.16 #define HAVE_NAMESPACES

12.17.1.17 #define HAVE_NETCDF 1

12.17.1.18 #define HAVE_POW 1

12.17.1.19 #define HAVE_PROJ4 1

12.17.1.20 #define HAVE_SQRT 1

12.17.1.21 #define HAVE_STD

12.17.1.22 #define HAVE_STDBOOL_H 1

12.17.1.23 #define HAVE_STDINT_H 1

12.17.1.24 #define HAVE_STDLIB_H 1

12.17.1.25 #define HAVE_STL

12.17.1.26 #define HAVE_STRING_H 1

12.17.1.27 #define HAVE_STRINGS_H 1

12.17.1.28 #define HAVE_STRSTR 1

12.17.1.29 #define HAVE_SYS_STAT_H 1

12.17.1.30 #define HAVE_SYS_TYPES_H 1

12.18 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"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::Data

Functions

- boost::shared_ptr< Data > MetNoFimex::createData (CDMDataType datatype, size_t length) throw (CDMException)

  create a Data-pointer of the datatype

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

  create a dataslice from another Data object

12.19 include/fimex/DataImpl.h File Reference

```
#include <typeinfo>
#include <boost/shared_ptr.hpp>
#include <string>
#include <sstream>
#include <iostream>
#include <cmath>
#include "fimex/Data.h"
#include "fimex/CDMDataType.h"
#include "fimex/CDMException.h"
#include "fimex/Utils.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::DataImpl< C >

Functions

- template<typename T1, typename T2>
  boost::shared_array< T1 > MetNoFimex::duplicateArrayType (const boost::shared_array< T2 > &inData, long length)

  create a new shared array with a different type using static_cast

- template<typename T1, typename T2>
  const boost::shared_array< T1 > MetNoFimex::constConvertArrayType (const boost::shared_array< T2 > &inData, long length)

  return a shared array of this data, possibly pointer to internal data

- template<class InputIterator>
  boost::shared_ptr< Data > MetNoFimex::createData (CDMDataType datatype, size_t length, InputIterator first, InputIterator last) throw (CDMException)

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

- template<typename C>
  void MetNoFimex::recursiveCopyMultiDimData (C **orgData, C **newData, const **std::vector**< size_t > &orgDimSize, const **std::vector**< size_t > &orgSliceSize, const **std::vector**< size_t > &newStart, const **std::vector**< size_t > &newSize, size_t currentDim)
- template<typename T1, typename T2>
  boost::shared_array< T1 > MetNoFimex::convertArrayType (const boost::shared_array< T2 > &inData, size_t length, double oldFill, double oldScale, double oldOffset, double newFill, double newScale, double newOffset)

12.20 include/fimex/DataTypeChanger.h File Reference

```
#include "fimex/CDMDataType.h"
#include "boost/shared_ptr.hpp"
#include "fimex/CDMException.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::DataTypeChanger

12.21 include/fimex/Felt_Array.h File Reference

```
#include <string>
#include <set>
#include <vector>
#include <map>
#include <ctime>
#include <boost/array.hpp>
#include "fimex/Felt_File_Error.h"
```

Namespaces

- namespace MetNoFelt

Classes

- struct MetNoFelt::ShortPairLess
- class MetNoFelt::Felt_Array

  encapsulate parameters of a felt file

Typedefs

- typedef set< pair< short, short >, ShortPairLess > MetNoFelt::ShortPairSet
- typedef map< pair< short, short >, short, ShortPairLess > MetNoFelt::ShortPairMap

Functions

- time_t MetNoFelt::index16toTime (const boost::array< short, 16 > &idx)
- pair< short, short > MetNoFelt::index16toLevelPair (const boost::array< short, 16 > &idx)

12.22 include/fimex/Felt_File.h File Reference

```
#include <ctime>
#include <map>
#include <vector>
#include <string>
#include <boost/shared_ptr.hpp>
#include <boost/shared_array.hpp>
#include "fimex/Data.h"
#include "fimex/Felt_Array.h"
#include "fimex/Felt_File_Error.h"
#include "fimex/FeltParameters.h"
#include "fimex/Logger.h"
```

Namespaces

- namespace MetNoFelt

Classes

- class MetNoFelt::Felt_File

  Felt File access.

12.23 include/fimex/Felt_File_Error.h File Reference

```
#include <exception>
#include <string>
```

Namespaces

- namespace MetNoFelt

Classes

- class MetNoFelt::Felt_File_Error

12.24 include/fimex/FeltCDMReader.h File Reference

```
#include <string>
#include <vector>
#include <map>
#include <boost/shared_ptr.hpp>
#include "fimex/Felt_File.h"
#include "fimex/CDMReader.h"
#include "fimex/CDMDimension.h"
#include "fimex/ReplaceStringObject.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::FeltCDMReader

12.25 include/fimex/FeltParameters.h File Reference

```
#include <map>
#include <string>
#include <vector>
#include <boost/array.hpp>
#include "fimex/Felt_File_Error.h"
```

Namespaces

- namespace MetNoFelt

Classes

- class MetNoFelt::FeltParameters

Functions

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

12.26 include/fimex/GribApiCDMWriter.h File Reference

```
#include "fimex/CDMWriter.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::GribApiCDMWriter

12.27 include/fimex/GribApiCDMWriter_Impl1.h File Reference

```
#include "fimex/GribApiCDMWriter_ImplAbstract.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::GribApiCDMWriter_Impl1

12.28 include/fimex/GribApiCDMWriter_Impl2.h File Reference

```
#include "fimex/GribApiCDMWriter_ImplAbstract.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::GribApiCDMWriter_Impl2

12.29 include/fimex/GribApiCDMWriter_ImplAbstract.h File Reference

```
#include <vector>
#include <grib_api.h>
#include <fstream>
#include <iostream>
#include "fimex/Logger.h"
#include "fimex/CDMWriter.h"
#include "fimex/XMLDoc.h"
#include "fimex/CDMException.h"
#include "fimex/TimeUnit.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::GribApiCDMWriter_ImplAbstract

12.30 include/fimex/interpolation.h File Reference

```
#include <proj_api.h>
#include <math.h>
```

Defines

- #define MIFI_NEAREST_NEIGHBOR 0

  interpolation method

- #define MIFI_BILINEAR 1

  interpolation method

- #define MIFI_BICUBIC 2

  interpolation method

- #define MIFI_VECTOR_KEEP_SIZE 0

  vector projection flag

- #define MIFI_VECTOR_RESIZE 1

  vector projection flag

- #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

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

- void 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_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_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)

- size_t mifi_bad2nanf (float *posPtr, float *endPtr, float badVal)
- size_t mifi_nanf2bad (float *posPtr, float *endPtr, float badVal)

12.30.1 Define Documentation

12.30.1.1 #define MIFI_BICUBIC 2

interpolation method

flag for bicubic interpolation

Warning:

not implemented yet

12.30.1.2 #define MIFI_BILINEAR 1

interpolation method

flag for bilinear interpolation

12.30.1.3 #define MIFI_DEBUG 0

debug flag

12.30.1.4 #define MIFI_ERROR -1

return code, error

12.30.1.5 #define MIFI_LATITUDE 2

latitude projection axis in degrees

12.30.1.6 #define MIFI_LONGITUDE 1

longitude projection axis in degrees

12.30.1.7 #define MIFI_NEAREST_NEIGHBOR 0

interpolation method

flag for nearest neighbor interpolation

12.30.1.8 #define MIFI_OK 1

return code, ok

12.30.1.9 #define MIFI_PROJ_AXIS 0

projection axis in m-equivalent

12.30.1.10 #define MIFI_UNDEFINED_D (nan(""))

undefined value for doubles

12.30.1.11 #define MIFI_UNDEFINED_F (nanf(""))

undefined value for floats

12.30.1.12 #define MIFI_VECTOR_KEEP_SIZE 0

vector projection flag

new size will be like old size

12.30.1.13 #define MIFI_VECTOR_RESIZE 1

vector projection flag

vector might change size with projection

12.30.2 Function Documentation

12.30.2.1 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

12.30.2.2 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

12.30.2.3 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

See also:

http://java.sun.com/products/java-media/jai/forDevelopers/jai-apidocs/javax/media/

12.30.2.4 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

12.30.2.5 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

12.30.2.6 void 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 * (in(a) - in(b)) / (a - b)

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

12.30.2.7 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

12.30.2.8 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

12.30.2.9 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_NEAREST_NEIGHBOR MIFI_BILINEAR MIFI_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

12.30.2.10 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

12.30.2.11 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

12.30.2.12 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

12.30.2.13 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

12.30.2.14 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

12.31 include/fimex/Logger.h File Reference

```
#include <boost/shared_ptr.hpp>
#include <string>
#include <sstream>
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::Logger

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)

12.31.1 Define Documentation

12.31.1.1 #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

12.32 include/fimex/NetCDF_CDMWriter.h File Reference

```
#include "fimex/CDMWriter.h"
#include "fimex/config.h"
#include <map>
#include <string>
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::NetCDF_CDMWriter

12.33 include/fimex/NetCDF_CF10_CDMReader.h File Reference

```
#include "fimex/config.h"
#include "fimex/CDMReader.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::NetCDF_CF10_CDMReader

12.34 include/fimex/NetCDF_Utils.h File Reference

```
#include <boost/shared_ptr.hpp>
#include "fimex/config.h"
#include "fimex/CDMDataType.h"
#include "fimex/Data.h"
```

Namespaces

- namespace MetNoFimex

Functions

- NcType MetNoFimex::cdmDataType2ncType (CDMDataType dt)
- CDMDataType MetNoFimex::ncType2cdmDataType (NcType dt)
- boost::shared_ptr< Data > MetNoFimex::ncValues2Data (NcValues *values, NcType dt, size_t length)

12.35 include/fimex/Null_CDMWriter.h File Reference

```
#include "fimex/CDMWriter.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::Null_CDMWriter

12.36 include/fimex/ReplaceStringObject.h File Reference

```
#include <iostream>
#include <string>
#include <vector>
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::ReplaceStringObject

12.37 include/fimex/ReplaceStringTimeObject.h File Reference

```
#include "fimex/ReplaceStringObject.h"
#include <ctime>
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::ReplaceStringTimeObject

12.38 include/fimex/TimeLevelDataSliceFetcher.h File Reference

```
#include <boost/shared_ptr.hpp>
#include "fimex/CDMReader.h"
#include "fimex/Data.h"
#include "fimex/Logger.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::TimeLevelDataSliceFetcher

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

12.39 include/fimex/TimeSpec.h File Reference

```
#include "fimex/TimeUnit.h"
#include "fimex/CDMException.h"
#include <vector>
#include <string>
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::TimeSpec

12.40 include/fimex/TimeUnit.h File Reference

```
#include "boost/shared_ptr.hpp"
#include "fimex/Units.h"
#include "fimex/CDMException.h"
#include <iostream>
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::FimexTime
- class MetNoFimex::TimeUnit

Functions

- **std::ostream** & MetNoFimex::operator<< (**std::ostream** &out, const FimexTime &fTime)
- FimexTime MetNoFimex::string2FimexTime (const **std::string** &str) throw (CDMException)

12.41 include/fimex/Units.h File Reference

```
#include <string>
#include "fimex/CDMException.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::UnitException
- class MetNoFimex::Units

Functions

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

12.42 include/fimex/Utils.h File Reference

```
#include <vector>
#include <sstream>
```

Namespaces

- namespace MetNoFimex

Functions

- int MetNoFimex::round (double num)
- **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<typename T>
  T MetNoFimex::string2type (**std::string** s)

12.43 include/fimex/XMLDoc.h File Reference

```
#include <boost/utility.hpp>
#include <boost/shared_ptr.hpp>
#include <string>
#include <libxml/tree.h>
#include <libxml/xpath.h>
#include "fimex/CDMException.h"
#include "fimex/XMLDoc.h"
```

Namespaces

- namespace MetNoFimex

Classes

- class MetNoFimex::XMLDoc

Typedefs

- 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)

Index