NMPB08 – Railway Emission Model

Reference Manual

Van Maercke Dirk
CSTB Grenoble
19/07/2011

The rest of this file was generated using Doxygen 1.7.4

Table of Contents

NMPB08 Railway Emission Model 2
Public functions implemented in the RailwayEmissionNMPB08 library 2
Loading and consulting the train database 2
Defining a railway traffic 2
Getting the equivalent source model 2
Setting and getting options 2
Class Index 4
File Index 5
Class Documentation 6
RailwaySourceModel::InvalidHeight 6
RailwayDatabase 7
RailwayElementarySource 10
RailwayEmission 12
RailwayEntity 14
RailwayEquivalentSource 16
RailwaySource 17
RailwaySourceModel 19
RailwaySourcePosition 23
RailwayTraffic 25
RailwayTrafficComponent 27
RailwayTrain 28
RailwayTrainUnit 30
RailwayUnit 31
ScreenBodyInteraction 33
Spectrum 35
File Documentation 36
Doxyfile.dox 36
Linux/RailwayEmissionNMPB08.h 37
MingW/RailwayEmissionNMPB08.h 48
RailwayEmissionNMPB08.h 59
RailwayDatabase.cpp 70
RailwayDatabase.h 73
RailwayEmission.cpp 75
RailwayEmission.h 80
RailwayEmissionNMPB.cpp 81
stdafx.cpp 82
stdafx.h 83
Index 84

NMPB08 Railway Emission Model

The railway emission model is defined in NF S31 133:2011. The standard defines the source model in terms of souce heights and directivity but does not include actual data for the associated sound powers.

The sound power calculations in this library are based on a XML-based database format defined by CSTB, RFF and SNCF.

Actual data is available from RFF.

Public functions implemented in the RailwayEmissionNMPB08 library

Loading and consulting the train database

- NMPB08_LoadRailwayDatabase
- NMPB08_DumpRailwayDatabase
- NMPB08_EnumRailwayDatabase

Defining a railway traffic

- NMPB08_CreateRailwayTraffic
- NMPB08_DeleteRailwayTraffic
- NMPB08_ClearRailwayTraffic
- NMPB08_AddRailwayTraffic

Getting the equivalent source model

- NMPB08_SetRailwayEmissionAngles
- NMPB08_SetRailCorrection
- NMPB08_GetRailCorrection
- NMPB08_GetRailwayEmission
- NMPB08_EnumRailwaySources

Setting and getting options

- NMPB08_SetRailwayOptions
- NMPB08_GetRailwayOptions

Class Index

Class List

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

RailwaySourceModel::InvalidHeight (Exception thrown when an invalid source is detected and **OPTION_MODIFY_HEIGHTS** is not set) 6

RailwayDatabase (Database of railway rolling stock) 7

RailwayElementarySource (Structure used for enumerating the equivalent acoustical sources associated with a train or a unit) 10

RailwayEmission (Emission associated with a railway traffic is represented as a set of elementary point sources with associated sound power and directivity) 12

RailwayEntity (Structure used for enumerating the contents of the database) 14

RailwayEquivalentSource (Elementary emission data represents apparent sound power for an equivalent line source emitted in a given direction) 16

RailwaySource (Level 1 of the database source records describe elementary sources and their sound power as a function of operating conditions) 17

RailwaySourceModel (Railway source model links traffic data to the sound power radiated from equivalent sources) 19

RailwaySourcePosition (Auxiliary structure for assigning and positionning an elementary source on a railway unit) 23

RailwayTraffic 25

RailwayTrafficComponent (Single railway traffic component. Each component is defined as a number of units (or trains) of a given type that ciruclate at the same speed on the same track, the same speed. A complete traffic may be composed of one or more components) 27

RailwayTrain (Level 3 of the database train records describe complete trains made up of one or more units) 28

RailwayTrainUnit (Auxiliary structure for assigning sequences of units to trains) 30

RailwayUnit (Level 2 of the database unit records describe individual rolling stock units) 31

ScreenBodyInteraction 33

Spectrum 35

File Index

File List

Here is a list of all files with brief descriptions:

RailwayDatabase.cpp 70
RailwayDatabase.h 73
RailwayEmission.cpp 75
RailwayEmission.h 80
RailwayEmissionNMPB.cpp 81
RailwayEmissionNMPB08.h 59
stdafx.cpp 82
stdafx.h 83
Linux/RailwayEmissionNMPB08.h 37
MingW/RailwayEmissionNMPB08.h 48

Class Documentation

RailwaySourceModel::InvalidHeight Class Reference

exception thrown when an invalid source is detected and OPTION_MODIFY_HEIGHTS is not set

`#include <RailwayEmission.h>`

Detailed Description

exception thrown when an invalid source is detected and OPTION_MODIFY_HEIGHTS is not set

Definition at line 116 of file RailwayEmission.h.

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

- RailwayEmission.h

RailwayDatabase Class Reference

the database of railway rolling stock
`#include <RailwayDatabase.h>`

Public Member Functions

- RailwayDatabase (void)
- int LoadFile (const char *filename, bool trace=false)
  load an external file into the database
- int GetErrorCode (void)
  get the most recent error code
- int Dump (const char *filename)
  dump the contents of the database to a text file
- int EnumEntities (EnumRailwayEntities enumProc, unsigned int include_types, void *userdata)
  enumerates the contents of the database
- int EnumSources (EnumRailwaySources enumProc, IDREF id, void *userdata, double pos=0.0)
  enumerates the elementary sources associated with a unit or a train
- RailwaySource const * getSource (IDREF ref)
  get the elementary source record
- RailwayUnit const * getUnit (IDREF ref)
  get the rolling stock unit record
- RailwayTrain const * getTrain (IDREF ref)
  get the train record

Detailed Description

the database of railway rolling stock

the database is structures at three levels : train, units and elementary sources

- trains are sequences composed of units
- units are defined in terms of equivalent sources
- elementary sources are described in terms of sound power and direcitivity

a railway traffic may be composed from trains and/or units.

Definition at line 147 of file RailwayDatabase.h.

Constructor & Destructor Documentation

RailwayDatabase::RailwayDatabase (void) `[inline]`

Definition at line 151 of file RailwayDatabase.h.

Member Function Documentation

int RailwayDatabase::Dump (const char *_filename_)

dump the contents of the database to a text file

Parameters:

filename	name of the text file. If the file name is a NULL pointer or points to an empty string, the output is written to stdout.

Definition at line 485 of file RailwayDatabase.cpp.

int RailwayDatabase::EnumEntities (EnumRailwayEntities_enumProc_, unsigned int_include_types_, void *_userdata_)

enumerates the contents of the database

Parameters:

enumProc	a user defined callback function, called once for each selected entry in the database
include_types	type of database entries to be enumerated
userdata	user defined data to be passed to the callback function

Definition at line 524 of file RailwayDatabase.cpp.

int RailwayDatabase::EnumSources (EnumRailwaySources_enumProc_, IDREF_id_, void *_userdata_, double_pos_ = `0.0`)

enumerates the elementary sources associated with a unit or a train

Parameters:

enumProc	a user defined callback function, called once for each elementary sources
id	identification code for train or unit
userdata	user defined data to be passed to the callback function
pos	position along the train (for internal use only)

Definition at line 583 of file RailwayDatabase.cpp.

int RailwayDatabase::GetErrorCode (void) `[inline]`

get the most recent error code

Definition at line 163 of file RailwayDatabase.h.

RailwaySource const * RailwayDatabase::getSource (IDREF_ref_)

get the elementary source record

Parameters:

ref	the identifier of the elementary source

Definition at line 223 of file RailwayDatabase.cpp.

RailwayTrain const * RailwayDatabase::getTrain (IDREF*ref*)

get the train record

Parameters:

ref	the identifier of the train

Definition at line 241 of file RailwayDatabase.cpp.

RailwayUnit const * RailwayDatabase::getUnit (IDREF*ref*)

get the rolling stock unit record

Parameters:

ref	the identifier of the rolling stock unit

Definition at line 232 of file RailwayDatabase.cpp.

int RailwayDatabase::LoadFile (const char **filename*, bool*trace* = `false`)

load an external file into the database

Parameters:

filename	name of the file
trace	if true, trace operations to stdout

Returns:

0 if successful, an error code otherwise

Definition at line 144 of file RailwayDatabase.cpp.

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

- RailwayDatabase.h
- RailwayDatabase.cpp

RailwayElementarySource Struct Reference

structure used for enumerating the equivalent acoustical sources associated with a train or a unit

`#include <RailwayEmissionNMPB08.h>`

Public Attributes

- const char * id
- double pos
- double height
- double vref
- double slope
- int hdir_model
- int vdir_model
- double const * Lw

Detailed Description

structure used for enumerating the equivalent acoustical sources associated with a train or a unit

Definition at line 430 of file RailwayEmissionNMPB08.h.

Member Data Documentation

int RailwayElementarySource::hdir_model

horizontal directivity (see NMPB_RailwayEmission_HorizontalDirectivity)

Definition at line 437 of file RailwayEmissionNMPB08.h.

double RailwayElementarySource::height

height of the equivalent point source

Definition at line 434 of file RailwayEmissionNMPB08.h.

const char * RailwayElementarySource::id

identifier

Definition at line 432 of file RailwayEmissionNMPB08.h.

double const * RailwayElementarySource::Lw

sound power spectrum at the reference speed

Definition at line 439 of file RailwayEmissionNMPB08.h.

double RailwayElementarySource::pos

position of the source along the train

Definition at line 433 of file RailwayEmissionNMPB08.h.

double RailwayElementarySource::slope

speed dependency of the sound power

Definition at line 436 of file RailwayEmissionNMPB08.h.

int RailwayElementarySource::vdir_model

vertical directivity (see NMPB_RailwayEmission_VerticalDirectivity)

Definition at line 438 of file RailwayEmissionNMPB08.h.

double RailwayElementarySource::vref

reference speed

Definition at line 435 of file RailwayEmissionNMPB08.h.

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

- Linux/RailwayEmissionNMPB08.h
- MingW/RailwayEmissionNMPB08.h
- RailwayEmissionNMPB08.h

RailwayEmission Struct Reference

the emission associated with a railway traffic is represented as a set of elementary point sources with associated sound power and directivity

`#include <RailwayEmissionNMPB08.h>`

Public Attributes

- unsigned int nbFreq
- double * freq
- double sin_h
- double sin_v
- unsigned int nbSources
- RailwayEquivalentSource * source

Detailed Description

the emission associated with a railway traffic is represented as a set of elementary point sources with associated sound power and directivity

Note:

the horizontal angle is coded as the sinus of the angle between the direction of propagation and the vertical plane perpendicular to the track
the vertical angle is coded as the sinus of the angle between the direction of propagation and the horizontal plane containing the track

Definition at line 156 of file RailwayEmissionNMPB08.h.

Member Data Documentation

double * RailwayEmission::freq

table of frequency bands

Definition at line 159 of file RailwayEmissionNMPB08.h.

unsigned int RailwayEmission::nbFreq

number of frequency bands

Definition at line 158 of file RailwayEmissionNMPB08.h.

unsigned int RailwayEmission::nbSources

number of equivalent sources

Definition at line 162 of file RailwayEmissionNMPB08.h.

double RailwayEmission::sin_h

horizontal component of the direction of propagation

Definition at line 160 of file RailwayEmissionNMPB08.h.

double RailwayEmission::sin_v

vertical component of the direction of propagation

Definition at line 161 of file RailwayEmissionNMPB08.h.

RailwayEquivalentSource * RailwayEmission::source

table of equivalent sources

Definition at line 163 of file RailwayEmissionNMPB08.h.

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

- Linux/RailwayEmissionNMPB08.h
- MingW/RailwayEmissionNMPB08.h
- RailwayEmissionNMPB08.h

RailwayEntity Struct Reference

structure used for enumerating the contents of the database
`#include <RailwayEmissionNMPB08.h>`

Public Attributes

- unsigned int type
- const char * id
- const char * name
- const char * image
- double length
- double vmax

Detailed Description

structure used for enumerating the contents of the database

Definition at line 255 of file RailwayEmissionNMPB08.h.

Member Data Documentation

const char * RailwayEntity::id

identifier

Definition at line 258 of file RailwayEmissionNMPB08.h.

const char * RailwayEntity::image

name of image file

Definition at line 260 of file RailwayEmissionNMPB08.h.

double RailwayEntity::length

lenght in meters

Definition at line 261 of file RailwayEmissionNMPB08.h.

const char * RailwayEntity::name

full name

Definition at line 259 of file RailwayEmissionNMPB08.h.

unsigned int RailwayEntity::type

unit, train or part of a train

Definition at line 257 of file RailwayEmissionNMPB08.h.

double RailwayEntity::vmax

maximum speed in km/h

Definition at line 262 of file RailwayEmissionNMPB08.h.

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

- Linux/RailwayEmissionNMPB08.h
- MingW/RailwayEmissionNMPB08.h
- RailwayEmissionNMPB08.h

RailwayEquivalentSource Struct Reference

elementary emission data represents apparent sound power for an equivalent line source emitted in a given direction

`#include <RailwayEmissionNMPB08.h>`

Public Attributes

- double height
- bool has_Lw
- double * Lw_dir

Detailed Description

elementary emission data represents apparent sound power for an equivalent line source emitted in a given direction

Definition at line 139 of file RailwayEmissionNMPB08.h.

Member Data Documentation

bool RailwayEquivalentSource::has_Lw

true if the equivalent source has a finite sound power

Definition at line 142 of file RailwayEmissionNMPB08.h.

double RailwayEquivalentSource::height

height of the equivalent point source

Definition at line 141 of file RailwayEmissionNMPB08.h.

double * RailwayEquivalentSource::Lw_dir

sound power per third octave band

Definition at line 143 of file RailwayEmissionNMPB08.h.

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

- Linux/RailwayEmissionNMPB08.h
- MingW/RailwayEmissionNMPB08.h
- RailwayEmissionNMPB08.h

RailwaySource Struct Reference

level 1 of the database source records describe elementary sources and their sound power as a function of operating conditions

`#include <RailwayDatabase.h>`

Public Member Functions

- RailwaySource (void)

Public Attributes

- double height
- double vref
- double slope
- int hdir_model
- int vdir_model
- Spectrum Lw

Detailed Description

level 1 of the database source records describe elementary sources and their sound power as a function of operating conditions

Definition at line 60 of file RailwayDatabase.h.

Constructor & Destructor Documentation

RailwaySource::RailwaySource (void) [inline]

Definition at line 69 of file RailwayDatabase.h.

Member Data Documentation

int RailwaySource::hdir_model

horizontal directivity (see NMPB_RailwayEmission_HorizontalDirectivity)

Definition at line 65 of file RailwayDatabase.h.

double RailwaySource::height

height of the equivalent source

Definition at line 62 of file RailwayDatabase.h.

Spectrum RailwaySource::Lw

sound power spectrum at the reference speed

Definition at line 67 of file RailwayDatabase.h.

double RailwaySource::slope

speed dependency of sound power

Definition at line 64 of file RailwayDatabase.h.

int RailwaySource::vdir_model

vertical directivity (see NMPB_RailwayEmission_VerticalDirectivity)

Definition at line 66 of file RailwayDatabase.h.

double RailwaySource::vref

reference speed

Definition at line 63 of file RailwayDatabase.h.

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

- RailwayDatabase.h

RailwaySourceModel Class Reference

the railway source model links traffic data to the sound power radiated from equivalent sources
`#include <RailwayEmission.h>`

Classes

- class InvalidHeight
- *exception thrown when an invalid source is detected and OPTION_MODIFY_HEIGHTS is not set* struct **SourceDistribution**

distribution of sound power over different source heights Public Member Functions

- RailwaySourceModel (void)
  constructor
- virtual ~RailwaySourceModel (void)
  destructor
- bool CreateTraffic (double hours)
  initialise the traffic record
- bool ClearTraffic (void)
  clears the traffic flow
- bool SetRailCorrection (NMPB_RailCorrectionType corr)
  set the rail correction
- NMPB_RailCorrectionType GetRailCorrection (void)
  get the rail correction
- bool AddTraffic (const char *name, double nb, double speed)
  add trains or units to the traffic flow
- bool SetEmissionAngles (double sin_h, double sin_v)
  defines the angles of emission for the calculation of apparent sound power
- RailwayEmission const * GetEmission (ScreenBodyInteraction *screenBodyInteraction)
  calculates and returns the corresponding emission model in terms of equivalent source positions and (apparent) sound power
- RailwayTraffic const * GetTraffic (void)
  return the current traffic data associated with the model
- unsigned int SetOptions (unsigned int option, bool on_off)
  enables or disables selected options
- unsigned int GetOptions (unsigned int option)
  return the currently selected options

Detailed Description

the railway source model links traffic data to the sound power radiated from equivalent sources

Definition at line 22 of file RailwayEmission.h.

Constructor & Destructor Documentation

RailwaySourceModel::RailwaySourceModel (void) [inline]

constructor

Definition at line 29 of file RailwayEmission.h.

virtual RailwaySourceModel::~RailwaySourceModel (void) [inline, virtual]

destructor

Definition at line 41 of file RailwayEmission.h.

Member Function Documentation

bool RailwaySourceModel::AddTraffic (const char *_name_, double _nb_, double _speed_)

add trains or units to the traffic flow

- name identification of the train or unit as specified in the database
- nb number of movements over the observation period
- speed speed of the trains in km/h

Definition at line 123 of file RailwayEmission.cpp.

bool RailwaySourceModel::ClearTraffic (void) [inline]

clears the traffic flow

Definition at line 62 of file RailwayEmission.h.

bool RailwaySourceModel::CreateTraffic (double _hours_) [inline]

initialise the traffic record

- hours duration of the period of obervation in hours

Definition at line 52 of file RailwayEmission.h.

RailwayEmission const * RailwaySourceModel::GetEmission (ScreenBodyInteraction *_screenBodyInteraction_)

calculates and returns the corresponding emission model in terms of equivalent source positions and (apparent) sound power

Returns:

pointer to the emission model, null pointer in case of error

Note:

in case the traffic refers trains and units with sources defined at heights other than those of the emission model, an error willbe detected and the function will return a NULL pointer. This behaviour can be avoided by setting the OPTION_MODIFY_HEIGHTS option.

Definition at line 148 of file RailwayEmission.cpp.

unsigned int RailwaySourceModel::GetOptions (unsigned int*option*) [inline]

return the currently selected options

Parameters:

option	options to be reported

Returns:

the selected set of options

Definition at line 168 of file RailwayEmission.h.

NMPB_RailCorrectionType RailwaySourceModel::GetRailCorrection (void) [inline]

get the rail correction

Definition at line 81 of file RailwayEmission.h.

RailwayTraffic const* RailwaySourceModel::GetTraffic (void) [inline]

return the current traffic data associated with the model

Returns:

pointer to the traffic data, null pointer in case of error

Definition at line 138 of file RailwayEmission.h.

bool RailwaySourceModel::SetEmissionAngles (double*sin_h*, double*sin_v*) [inline]

defines the angles of emission for the calculation of apparent sound power

- sin_h sinus of the horizontal angle, 0 = plane perpendicular to the track
- sin_v sinus of the vertical angle, 0 = horizontal plane

Definition at line 104 of file RailwayEmission.h.

unsigned int RailwaySourceModel::SetOptions (unsigned int*option*, bool*on_off*) [inline]

enables or disables selected options

Parameters:

option	options to be set or cleared
on_off	if true, options will be set, otherwise options will be cleared

Returns:

the modified set of options

Definition at line 152 of file RailwayEmission.h.

bool RailwaySourceModel::SetRailCorrection (NMPB_RailCorrectionType *corr*) [inline]

set the rail correction

Definition at line 72 of file RailwayEmission.h.

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

- RailwayEmission.h
- RailwayEmission.cpp

RailwaySourcePosition Struct Reference

auxiliary structure for assigning and positionning an elementary source on a railway unit

`#include <RailwayDatabase.h>`

Public Member Functions

- RailwaySourcePosition (void)

Public Attributes

- IDREF ref
- int nb
- double pos
- double spacing

Detailed Description

auxiliary structure for assigning and positionning an elementary source on a railway unit

Definition at line 80 of file RailwayDatabase.h.

Constructor & Destructor Documentation

RailwaySourcePosition::RailwaySourcePosition (void) `[inline]`

Definition at line 87 of file RailwayDatabase.h.

Member Data Documentation

int RailwaySourcePosition::nb

number of elementary sources

Definition at line 83 of file RailwayDatabase.h.

double RailwaySourcePosition::pos

position of first source along the unit

Definition at line 84 of file RailwayDatabase.h.

IDREF RailwaySourcePosition::ref

identifier of elementary source

Definition at line 82 of file RailwayDatabase.h.

double RailwaySourcePosition::spacing

spacing of sources along the unit

Definition at line 85 of file RailwayDatabase.h.

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

- RailwayDatabase.h

RailwayTraffic Struct Reference

`#include <RailwayEmissionNMPB08.h>`

Public Attributes

- double nbHours
- NMPB_RailCorrectionType railCorrection
- unsigned int nbComponents
- unsigned int maxComponents
- RailwayTrafficComponent * component

Detailed Description

Note:

the period of reference is defined when the traffic is created, see NMPB08_CreateRailwayTraffic

railway traffic represents number of units (or trains) running on one track at specified speeds during a period of reference (in hours)

Definition at line 190 of file RailwayEmissionNMPB08.h.

Member Data Documentation

RailwayTrafficComponent * RailwayTraffic::component

table of traffic components

Definition at line 196 of file RailwayEmissionNMPB08.h.

unsigned int RailwayTraffic::maxComponents

for internal use only

Definition at line 195 of file RailwayEmissionNMPB08.h.

unsigned int RailwayTraffic::nbComponents

number of traffic components

Definition at line 194 of file RailwayEmissionNMPB08.h.

double RailwayTraffic::nbHours

duration of the period of reference in hours

Definition at line 192 of file RailwayEmissionNMPB08.h.

NMPB_RailCorrectionType RailwayTraffic::railCorrection

correction for rail and mounting conditions

Definition at line 193 of file RailwayEmissionNMPB08.h.

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

- Linux/RailwayEmissionNMPB08.h
- MingW/RailwayEmissionNMPB08.h
- RailwayEmissionNMPB08.h

RailwayTrafficComponent Struct Reference

a single railway traffic component. Each component is defined as a number of units (or trains) of a given type that ciruclate at the same speed on the same track, the same speed. A complete traffic may be composed of one or more components.

`#include <RailwayEmissionNMPB08.h>`

Public Attributes

- char * id
- double number
- double speed

Detailed Description

a single railway traffic component. Each component is defined as a number of units (or trains) of a given type that ciruclate at the same speed on the same track, the same speed. A complete traffic may be composed of one or more components.

Definition at line 173 of file RailwayEmissionNMPB08.h.

Member Data Documentation

char * RailwayTrafficComponent::id

identification of a unit or train

Definition at line 175 of file RailwayEmissionNMPB08.h.

double RailwayTrafficComponent::number

number of units/train per period of reference

Definition at line 176 of file RailwayEmissionNMPB08.h.

double RailwayTrafficComponent::speed

speed in km/h

Definition at line 177 of file RailwayEmissionNMPB08.h.

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

- Linux/RailwayEmissionNMPB08.h
- MingW/RailwayEmissionNMPB08.h
- RailwayEmissionNMPB08.h

RailwayTrain Struct Reference

level 3 of the database train records describe complete trains made up of one or more units

`#include <RailwayDatabase.h>`

Public Member Functions

- RailwayTrain (void)

Public Attributes

- std::string name
- std::string image
- std::vector< RailwayTrainUnit > unit

Detailed Description

level 3 of the database train records describe complete trains made up of one or more units

Definition at line 126 of file RailwayDatabase.h.

Constructor & Destructor Documentation

RailwayTrain::RailwayTrain (void) `[inline]`

Definition at line 132 of file RailwayDatabase.h.

Member Data Documentation

std::string RailwayTrain::image

name of the picture file

Definition at line 129 of file RailwayDatabase.h.

std::string RailwayTrain::name

identifier of the train

Definition at line 128 of file RailwayDatabase.h.

std::vector<RailwayTrainUnit> RailwayTrain::unit

list of units

Definition at line 130 of file RailwayDatabase.h.

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

- RailwayDatabase.h

RailwayTrainUnit Struct Reference

auxiliary structure for assigning sequences of units to trains

`#include <RailwayDatabase.h>`

Public Member Functions

- RailwayTrainUnit (void)

Public Attributes

- IDREF ref
- int nb

Detailed Description

auxiliary structure for assigning sequences of units to trains

Definition at line 113 of file RailwayDatabase.h.

Constructor & Destructor Documentation

RailwayTrainUnit::RailwayTrainUnit (void) `[inline]`

Definition at line 118 of file RailwayDatabase.h.

Member Data Documentation

int RailwayTrainUnit::nb

number of units

Definition at line 116 of file RailwayDatabase.h.

IDREF RailwayTrainUnit::ref

identifier of the unit of rolling stock

Definition at line 115 of file RailwayDatabase.h.

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

- RailwayDatabase.h

RailwayUnit Struct Reference

level 2 of the database unit records describe individual rolling stock units

`#include <RailwayDatabase.h>`

Public Member Functions

- RailwayUnit (void)

Public Attributes

- std::string name
- double length
- std::string image
- double vmax
- double cref
- bool part_of_train
- std::vector< RailwaySourcePosition > source

Detailed Description

level 2 of the database unit records describe individual rolling stock units

Definition at line 95 of file RailwayDatabase.h.

Constructor & Destructor Documentation

RailwayUnit::RailwayUnit (void) `[inline]`

Definition at line 105 of file RailwayDatabase.h.

Member Data Documentation

double RailwayUnit::cref

reflection coefficient for screen/train interaction

Definition at line 101 of file RailwayDatabase.h.

std::string RailwayUnit::image

name of the picture file

Definition at line 99 of file RailwayDatabase.h.

double RailwayUnit::length

lenght of the unit

Definition at line 98 of file RailwayDatabase.h.

std::string RailwayUnit::name

identifier of the unit of rolling stock

Definition at line 97 of file RailwayDatabase.h.

bool RailwayUnit::part_of_train

unit is not independent rolling stock

Definition at line 102 of file RailwayDatabase.h.

std::vector<RailwaySourcePosition> RailwayUnit::source

list of elementary sources

Definition at line 103 of file RailwayDatabase.h.

double RailwayUnit::vmax

maximum speed of the unit in km/h

Definition at line 100 of file RailwayDatabase.h.

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

- RailwayDatabase.h

ScreenBodyInteraction Struct Reference

`#include <RailwayEmissionNMPB08.h>`

Public Attributes

- int maxInteractions
- double dS
- double hS
- double dR
- double hR
- double * alpha

Detailed Description

configuration of interaction between the train body and a nearby screen

Definition at line 203 of file RailwayEmissionNMPB08.h.

Member Data Documentation

double * ScreenBodyInteraction::alpha

absoprtion applied to the inner side of the barrier

Definition at line 210 of file RailwayEmissionNMPB08.h.

double ScreenBodyInteraction::dR

horizontal distance from screen to receiver

Definition at line 208 of file RailwayEmissionNMPB08.h.

double ScreenBodyInteraction::dS

horizontal distance from source to screen

Definition at line 206 of file RailwayEmissionNMPB08.h.

double ScreenBodyInteraction::hR

height of receiver relative to the head of the track

Definition at line 209 of file RailwayEmissionNMPB08.h.

double ScreenBodyInteraction::hS

height of screen relative to the head of the track

Definition at line 207 of file RailwayEmissionNMPB08.h.

int ScreenBodyInteraction::maxInteractions

number of interactions to be taken into account

Definition at line 205 of file RailwayEmissionNMPB08.h.

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

- Linux/RailwayEmissionNMPB08.h
- MingW/RailwayEmissionNMPB08.h
- RailwayEmissionNMPB08.h

Spectrum Struct Reference

`#include <RailwayDatabase.h>`

Public Member Functions

- Spectrum (void)
- Spectrum (Spectrum const &other)

Detailed Description

Definition at line 39 of file RailwayDatabase.h.

Constructor & Destructor Documentation

Spectrum::Spectrum (void) [inline]

Definition at line 41 of file RailwayDatabase.h.

Spectrum::Spectrum (Spectrum const &*other*) [inline]

Definition at line 45 of file RailwayDatabase.h.

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

- RailwayDatabase.h

File Documentation

Doxyfile.dox File Reference

Linux/RailwayEmissionNMPB08.h File Reference

Classes

- struct RailwayEquivalentSource
- *elementary emission data represents apparent sound power for an equivalent line source emitted in a given direction* struct RailwayEmission
- *the emission associated with a railway traffic is represented as a set of elementary point sources with associated sound power and directivity* struct RailwayTrafficComponent
- *a single railway traffic component. Each component is defined as a number of units (or trains) of a given type that ciruclate at the same speed on the same track, the same speed. A complete traffic may be composed of one or more components.* struct RailwayTraffic
- struct ScreenBodyInteraction
- struct RailwayEntity
- *structure used for enumerating the contents of the database* struct RailwayElementarySource

structure used for enumerating the equivalent acoustical sources associated with a train or a unit Defines

- #define _COMPILE_NMPB extern "C"
  compiler specific options for creating shared libraries

Typedefs

- typedef double NMPB_RailCorrectionType
  correction for rail and mounting conditions as specified in the document "Production des cartes strategiques des grands axes routiers et ferrovaires", published by SETRA, August 2007.

  Note that the following values are indicative and not integral part of the method.

- typedef bool(* EnumRailwayEntities)(RailwayEntity const &info, void *userdata)
  user defined callback function for enumerating the contents of the database
- typedef bool(* EnumRailwaySources)(RailwayElementarySource const &info, void *userdata)
  user defined callback function for enumerating the equivalent acoustical sources associated with a train or a unit

Enumerations

- enum NPMP08_RailwayEmission_Error { ERROR_XML_PARSER = 1, ERROR_XML_SCHEMA = 2, ERROR_OPEN_FILE = 3, ERROR_INVALID_ID = 4, ERROR_INVALID_HEIGHT = 5, ERROR_XML_PARSER = 1, ERROR_XML_SCHEMA = 2, ERROR_OPEN_FILE = 3, ERROR_INVALID_ID = 4, ERROR_INVALID_HEIGHT = 5, ERROR_XML_PARSER = 1, ERROR_XML_SCHEMA = 2, ERROR_OPEN_FILE = 3, ERROR_INVALID_ID = 4, ERROR_INVALID_HEIGHT = 5 }
- *error types returned from the NMPB08_Railway software library* enum NMPB_RailwayEmission_HorizontalDirectivity { HDIR_OMNI = 0, HDIR_NMPB = 1, HDIR_OMNI = 0, HDIR_NMPB = 1, HDIR_OMNI = 0, HDIR_NMPB = 1 }
- *predefined horizontal directivity models* enum NMPB_RailwayEmission_VerticalDirectivity { VDIR_OMNI = 0, VDIR_NMPB = 1, VDIR_HEMI = 2, VDIR_OMNI = 0, VDIR_NMPB = 1, VDIR_HEMI = 2, VDIR_OMNI = 0, VDIR_NMPB = 1, VDIR_HEMI = 2 }

- *predefined vertical directivity models* enum NMPB_RailwayEmission_Options { OPTION_TRACE_DEBUG = 1, OPTION_NO_DIRECTIVITY = 2, OPTION_MODIFY_HEIGHTS = 4, OPTION_DEYGOUT_DIFFRACTION = 8, OPTION_TRACE_DEBUG = 1, OPTION_NO_DIRECTIVITY = 2, OPTION_MODIFY_HEIGHTS = 4, OPTION_DEYGOUT_DIFFRACTION = 8, OPTION_TRACE_DEBUG = 1, OPTION_NO_DIRECTIVITY = 2, OPTION_MODIFY_HEIGHTS = 4, OPTION_DEYGOUT_DIFFRACTION = 8 }
- *options that modify the behaviour of the calculation model* enum NMPB_RailwayEmission_Entities { ENTITY_UNIT = 1, ENTITY_TRAIN = 2, ENTITY_PARTIAL = 4, ENTITY_TRAFFIC = ENTITY_TRAIN | ENTITY_UNIT, ENTITY_ALL = ENTITY_TRAIN | ENTITY_UNIT | ENTITY_PARTIAL, ENTITY_UNIT = 1, ENTITY_TRAIN = 2, ENTITY_PARTIAL = 4, ENTITY_TRAFFIC = ENTITY_TRAIN | ENTITY_UNIT, ENTITY_ALL = ENTITY_TRAIN | ENTITY_UNIT | ENTITY_PARTIAL, ENTITY_UNIT = 1, ENTITY_TRAIN = 2, ENTITY_PARTIAL = 4, ENTITY_TRAFFIC = ENTITY_TRAIN | ENTITY_UNIT, ENTITY_ALL = ENTITY_TRAIN | ENTITY_UNIT | ENTITY_PARTIAL }

constants used for enumerating the contents of the database Functions

- _COMPILE_NMPB int NMPB08_LoadRailwayDatabase (const char *filename, bool log_stdout=false)
  loads the train database from an external file
- _COMPILE_NMPB int NMPB08_DumpRailwayDatabase (const char *filename=0)
  writes the contents of the database to a file
- _COMPILE_NMPB int NMPB08_EnumRailwayDatabase (EnumRailwayEntities enumProc, unsigned int include_types=ENTITY_TRAFFIC, void *userdata=0)
  enumerate the contents of the database
- _COMPILE_NMPB void * NMPB08_CreateRailwayTraffic (double nb_hours)
  create a context for converting railway traffic into an equivalent source model
- _COMPILE_NMPB int NMPB08_ClearRailwayTraffic (void *id)
  empties the traffic associated with the source model
- _COMPILE_NMPB int NMPB08_AddRailwayTraffic (void *id, const char *unit_or_train, double number, double speed)
  adds a traffic component to the source model
- _COMPILE_NMPB unsigned int NMPB08_SetRailwayOptions (void *id, unsigned int option, bool on_off)
  enables or disables selected options
- _COMPILE_NMPB unsigned int NMPB08_GetRailwayOptions (void *id, unsigned int option)
  return the currently selected options
- _COMPILE_NMPB unsigned int NMPB08_SetRailCorrection (void *id, NMPB_RailCorrectionType corr)
  set the rail correction
- _COMPILE_NMPB NMPB_RailCorrectionType NMPB08_GetRailCorrection (void *id)
  returns the current value of rail correction
- _COMPILE_NMPB RailwayEmission const * NMPB08_GetRailwayEmission (void *id, ScreenBodyInteraction *screenBodyInteraction=0)
  returns the equivalent source model for the current traffic state
- _COMPILE_NMPB int NMPB08_SetRailwayEmissionAngles (void *id, double sin_h, double sin_v)
  sets de emission angles for the equivalent source model
- _COMPILE_NMPB int NMPB08_DeleteRailwayTraffic (void *id)
  destroys the internal data structure used for converting railway traffic data into an equivalent source model.
- _COMPILE_NMPB int NMPB08_EnumRailwaySources (EnumRailwaySources enumProc, const char *name, void *userdata)
  enumerate the equivalent acoustical sources associated with a train or unit

Variables

- const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_BETON = 0.0
- const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_AUTRE = 3.0
- const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_BETON = 3.0
- const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_AUTRE = 6.0
- const NMPB_RailCorrectionType RAIL_ZONE_APPAREILS_VOIE = 6.0

Define Documentation

#define _COMPILE_NMPB extern "C"

compiler specific options for creating shared libraries

Definition at line 54 of file RailwayEmissionNMPB08.h.

Typedef Documentation

typedef bool(* EnumRailwayEntities)(RailwayEntity const &info, void *userdata)

user defined callback function for enumerating the contents of the database

Definition at line 269 of file RailwayEmissionNMPB08.h.

typedef bool(* EnumRailwaySources)(RailwayElementarySource const &info, void *userdata)

user defined callback function for enumerating the equivalent acoustical sources associated with a train or a unit

Definition at line 447 of file RailwayEmissionNMPB08.h.

typedef double NMPB_RailCorrectionType

correction for rail and mounting conditions as specified in the document "Production des cartes strategiques des grands axes routiers et ferrovaires", published by SETRA, August 2007.

Note that the following values are indicative and not integral part of the method.

- RAIL_LONG_TRAVERSES_BETON = 0 dB(A)
- RAIL_LONG_TRAVERSES_AUTRE = 3 dB(A)
- RAIL_COURT_TRAVERSES_BETON = 3 dB(A)
- RAIL_COURT_TRAVERSES_AUTRE = 6 dB(A)
-
- The end user may supply other values, e.g. for points and metal bridges

Definition at line 126 of file RailwayEmissionNMPB08.h.

Enumeration Type Documentation

enum NMPB_RailwayEmission_Entities

constants used for enumerating the contents of the database

Enumerator:

ENTITY_UNIT unit

ENTITY_TRAIN train

ENTITY_PARTIAL unit, only to be used as part of a train

ENTITY_TRAFFIC traffic units only

ENTITY_ALL all units and trains

ENTITY_UNIT unit

ENTITY_TRAIN train

ENTITY_PARTIAL unit, only to be used as part of a train

ENTITY_TRAFFIC traffic units only

ENTITY_ALL all units and trains

ENTITY_UNIT unit

ENTITY_TRAIN train

ENTITY_PARTIAL unit, only to be used as part of a train

ENTITY_TRAFFIC traffic units only

ENTITY_ALL all units and trains

Definition at line 242 of file RailwayEmissionNMPB08.h.

enum NMPB_RailwayEmission_HorizontalDirectivity

predefined horizontal directivity models

Enumerator:

HDIR_OMNI omnidirectionnel

HDIR_NMPB horizontal directivity as defined in NF S31-133

HDIR_OMNI omnidirectionnel

HDIR_NMPB horizontal directivity as defined in NF S31-133

HDIR_OMNI omnidirectionnel

HDIR_NMPB horizontal directivity as defined in NF S31-133

Definition at line 75 of file RailwayEmissionNMPB08.h.

enum NMPB_RailwayEmission_Options

options that modify the behaviour of the calculation model

Note:

OPTION_NO_DIRECTIVITY is useful if the directivity of the railway sources is integrated as part of the propagation model
OPTION_MODIFY_HEIGHTS is useful to adjust databases for different propagation models, i.e. when source heights in the database do not match those of the emission model

Enumerator:

OPTION_TRACE_DEBUG trace intermediate results to stdout

OPTION_NO_DIRECTIVITY do not include directivity

OPTION_MODIFY_HEIGHTS distribute sound powers over source heights

OPTION_DEYGOUT_DIFFRACTION use Deygout approximation for body/barrier interaction

OPTION_TRACE_DEBUG trace intermediate results to stdout

OPTION_NO_DIRECTIVITY do not include directivity

OPTION_MODIFY_HEIGHTS distribute sound powers over source heights

OPTION_DEYGOUT_DIFFRACTION use Deygout approximation for body/barrier interaction

OPTION_TRACE_DEBUG trace intermediate results to stdout

OPTION_NO_DIRECTIVITY do not include directivity

OPTION_MODIFY_HEIGHTS distribute sound powers over source heights

OPTION_DEYGOUT_DIFFRACTION use Deygout approximation for body/barrier interaction

Definition at line 103 of file RailwayEmissionNMPB08.h.

enum NMPB_RailwayEmission_VerticalDirectivity

predefined vertical directivity models

Enumerator:

VDIR_OMNI omnidirectional

VDIR_NMPB vertical directivity as defined in NF S31-133

VDIR_HEMI hemispheric emission upwards

VDIR_OMNI omnidirectional

VDIR_NMPB vertical directivity as defined in NF S31-133

VDIR_HEMI hemispheric emission upwards

VDIR_OMNI omnidirectional

VDIR_NMPB vertical directivity as defined in NF S31-133

VDIR_HEMI hemispheric emission upwards

Definition at line 85 of file RailwayEmissionNMPB08.h.

enum NPMP08_RailwayEmission_Error

error types returned from the NMPB08_Railway software library

Enumerator:

ERROR_XML_PARSER file could not be openened or parsed

ERROR_XML_SCHEMA incompatible XML file

ERROR_OPEN_FILE file could not be openened

ERROR_INVALID_ID identifier not defined in database

ERROR_INVALID_HEIGHT source height not supported in emission model

ERROR_XML_PARSER file could not be openened or parsed

ERROR_XML_SCHEMA incompatible XML file

ERROR_OPEN_FILE file could not be openened

ERROR_INVALID_ID identifier not defined in database

ERROR_INVALID_HEIGHT source height not supported in emission model

ERROR_XML_PARSER file could not be openened or parsed

ERROR_XML_SCHEMA incompatible XML file

ERROR_OPEN_FILE file could not be openened

ERROR_INVALID_ID identifier not defined in database

ERROR_INVALID_HEIGHT source height not supported in emission model

Definition at line 62 of file RailwayEmissionNMPB08.h.

Function Documentation

_COMPILE_NMPB int NMPB08_AddRailwayTraffic (void **id*, const char **unit_or_train*, double*number*, double*speed*)

adds a traffic component to the source model

Parameters:

id	handle to the internal structure used by the traffic model
unit_or_train	identification of the type of unit or train
number	number of units or trains passing during the reference period
speed	speed in km/h

Returns:

O if sucessful, an error code otherwise

Definition at line 681 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_ClearRailwayTraffic (void **id*)

empties the traffic associated with the source model

Parameters:

id	handle to the internal structure used by the traffic model

Returns:

O if sucessful, an error code otherwise

Definition at line 670 of file RailwayEmission.cpp.

_COMPILE_NMPB void* NMPB08_CreateRailwayTraffic (double*nb_hours*)

create a context for converting railway traffic into an equivalent source model

Parameters:

nb_hours	duration of the period of reference in hours

Returns:

a handle to an internal structure to be uses in consecutive calls to other function inside the library, a NULL pointer if error occured

Definition at line 659 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_DeleteRailwayTraffic (void **id*)

destroys the internal data structure used for converting railway traffic data into an equivalent source model.

Parameters:

id	handle to an internal data structure as returned by /ref NMPB08_CreateRailwayTraffic.

Returns:

0 if successful, an error code otherwise.

Definition at line 703 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_DumpRailwayDatabase (const char **filename* = `0`)

writes the contents of the database to a file

Parameters:

filename	the name of the output file in case the filename is a zero pointer or a zero length string, the output will be written to stdout

Returns:

0 if successful, otherwise an error code

Definition at line 16 of file RailwayDatabase.cpp.

_COMPILE_NMPB int NMPB08_EnumRailwayDatabase (EnumRailwayEntities*enumProc*, unsigned int*include_types* = `ENTITY_TRAFFIC`, void **userdata* = `0`)

enumerate the contents of the database

Parameters:

enumProc	user defined callback function
include_types	type of database entries to be enumerated
userdata	user defined data to be passed to the callback function

Definition at line 21 of file RailwayDatabase.cpp.

_COMPILE_NMPB int NMPB08_EnumRailwaySources (EnumRailwaySources *enumProc*, const char **name*, void **userdata*)

enumerate the equivalent acoustical sources associated with a train or unit

Parameters:

enumProc	user defined callback function
name	identification a train or unit
userdata	user defined data to be passed to the callback function

Definition at line 28 of file RailwayDatabase.cpp.

_COMPILE_NMPB NMPB_RailCorrectionType NMPB08_GetRailCorrection (void **id*)

returns the current value of rail correction

Parameters:

id	handle to the internal structure used by the traffic model

Returns:

the current value of the rail correction

_COMPILE_NMPB RailwayEmission const* NMPB08_GetRailwayEmission (void **id*, ScreenBodyInteraction **screenBodyInteraction* = 0)

returns the equivalent source model for the current traffic state

Parameters:

id	handle to the internal structure used by the traffic model
screenBodyInteraction	configuration of interaction between the train body and a nearby screen One interaction corresponds to a double reflection, once on the barrier, once on the car body. The equivalent reflection coefficient of the car bodies is encoded in the database and will modify the equivalent sound power of the image sources created through reflection. Absorption on the inner side of the barrier is also taken into account (see NF S 31-133, section 7.4.6).

Returns:

a pointer to the equivalent source model, a null pointer in case an invalid source height has been detected in the selected trains and units

Note:

if OPTION_H_INTERPOLATE is set, sources heights defined in the database will be automatically adapted to the 3 sources heights defined in the NMPB model. This is achieved by distributing the acoustical power over the predefined source heights proportional to the difference in height

Definition at line 692 of file RailwayEmission.cpp.

_COMPILE_NMPB unsigned int NMPB08_GetRailwayOptions (void *id, unsigned intoption)

return the currently selected options

Parameters:

id	handle to the internal structure used by the traffic model
option	options to be reported

Returns:

the selected set of options

Definition at line 726 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_LoadRailwayDatabase (const char *filename, boollog_stdout = false)

loads the train database from an external file

Parameters:

filename	name of the external file
log_stdout	if true, prints messages to stdout

Returns:

0 if successful, otherwise an error code

Definition at line 11 of file RailwayDatabase.cpp.

_COMPILE_NMPB unsigned int NMPB08_SetRailCorrection (void *id, NMPB_RailCorrectionTypecorr)

set the rail correction

Parameters:

id	handle to the internal structure used by the traffic model
corr	new correction for the rail / mounting condition

Returns:

O if sucessful, an error code otherwise

Definition at line 748 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_SetRailwayEmissionAngles (void *id, doublesin_h, doublesin_v)

sets de emission angles for the equivalent source model

Parameters:

id	handle to the internal structure used by the traffic model
sin_h	sinus of the angle of the propagation direction with the vertical plane perpendicular to the track
sin_v	sinus of the angle of the propagation direction with the horizontal plane containing the track

Returns:

0 if successful, an error code otherwise

Note:

if the OPTION_NO_DIRECIVITY is set, this function has no effect on the reported equivalent source model

Definition at line 737 of file RailwayEmission.cpp.

_COMPILE_NMPB unsigned int NMPB08_SetRailwayOptions (void *id, unsigned int option, bool on_off)

enables or disables selected options

Parameters:

id	handle to the internal structure used by the traffic model
option	options to be set or cleared
on_off	if true, options will be set, otherwise options will be cleared

Returns:

the modified set of options

Definition at line 715 of file RailwayEmission.cpp.

Variable Documentation

const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_AUTRE = 6.0

Definition at line 131 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_BETON = 3.0

Definition at line 130 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_AUTRE = 3.0

Definition at line 129 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_BETON = 0.0

Definition at line 128 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_ZONE_APPAREILS_VOIE = 6.0

Definition at line 132 of file RailwayEmissionNMPB08.h.

MingW/RailwayEmissionNMPB08.h File Reference

Classes

- struct RailwayEquivalentSource
- *elementary emission data represents apparent sound power for an equivalent line source emitted in a given direction* struct RailwayEmission
- *the emission associated with a railway traffic is represented as a set of elementary point sources with associated sound power and directivity* struct RailwayTrafficComponent
- *a single railway traffic component. Each component is defined as a number of units (or trains) of a given type that ciruclate at the same speed on the same track, the same speed. A complete traffic may be composed of one or more components.* struct RailwayTraffic
- struct ScreenBodyInteraction
- struct RailwayEntity
- *structure used for enumerating the contents of the database* struct RailwayElementarySource

structure used for enumerating the equivalent acoustical sources associated with a train or a unit Defines

- #define _COMPILE_NMPB extern "C"
  compiler specific options for creating shared libraries

Typedefs

- typedef double NMPB_RailCorrectionType
  correction for rail and mounting conditions as specified in the document "Production des cartes strategiques des grands axes routiers et ferrovaires", published by SETRA, August 2007.

  Note that the following values are indicative and not integral part of the method.

- typedef bool(* EnumRailwayEntities)(RailwayEntity const &info, void *userdata)
  user defined callback function for enumerating the contents of the database
- typedef bool(* EnumRailwaySources)(RailwayElementarySource const &info, void *userdata)
  user defined callback function for enumerating the equivalent acoustical sources associated with a train or a unit

Enumerations

- enum NPMP08_RailwayEmission_Error { ERROR_XML_PARSER = 1, ERROR_XML_SCHEMA = 2, ERROR_OPEN_FILE = 3, ERROR_INVALID_ID = 4, ERROR_INVALID_HEIGHT = 5, ERROR_XML_PARSER = 1, ERROR_XML_SCHEMA = 2, ERROR_OPEN_FILE = 3, ERROR_INVALID_ID = 4, ERROR_INVALID_HEIGHT = 5, ERROR_XML_PARSER = 1, ERROR_XML_SCHEMA = 2, ERROR_OPEN_FILE = 3, ERROR_INVALID_ID = 4, ERROR_INVALID_HEIGHT = 5 }
- *error types returned from the NMPB08_Railway software library* enum NMPB_RailwayEmission_HorizontalDirectivity { HDIR_OMNI = 0, HDIR_NMPB = 1, HDIR_OMNI = 0, HDIR_NMPB = 1, HDIR_OMNI = 0, HDIR_NMPB = 1 }
- *predefined horizontal directivity models* enum NMPB_RailwayEmission_VerticalDirectivity { VDIR_OMNI = 0, VDIR_NMPB = 1, VDIR_HEMI = 2, VDIR_OMNI = 0, VDIR_NMPB = 1, VDIR_HEMI = 2, VDIR_OMNI = 0, VDIR_NMPB = 1, VDIR_HEMI = 2 }

- *predefined vertical directivity models* enum NMPB_RailwayEmission_Options { OPTION_TRACE_DEBUG = 1, OPTION_NO_DIRECTIVITY = 2, OPTION_MODIFY_HEIGHTS = 4, OPTION_DEYGOUT_DIFFRACTION = 8, OPTION_TRACE_DEBUG = 1, OPTION_NO_DIRECTIVITY = 2, OPTION_MODIFY_HEIGHTS = 4, OPTION_DEYGOUT_DIFFRACTION = 8, OPTION_TRACE_DEBUG = 1, OPTION_NO_DIRECTIVITY = 2, OPTION_MODIFY_HEIGHTS = 4, OPTION_DEYGOUT_DIFFRACTION = 8 }
- *options that modify the behaviour of the calculation model* enum NMPB_RailwayEmission_Entities { ENTITY_UNIT = 1, ENTITY_TRAIN = 2, ENTITY_PARTIAL = 4, ENTITY_TRAFFIC = ENTITY_TRAIN | ENTITY_UNIT, ENTITY_ALL = ENTITY_TRAIN | ENTITY_UNIT | ENTITY_PARTIAL, ENTITY_UNIT = 1, ENTITY_TRAIN = 2, ENTITY_PARTIAL = 4, ENTITY_TRAFFIC = ENTITY_TRAIN | ENTITY_UNIT, ENTITY_ALL = ENTITY_TRAIN | ENTITY_UNIT | ENTITY_PARTIAL, ENTITY_UNIT = 1, ENTITY_TRAIN = 2, ENTITY_PARTIAL = 4, ENTITY_TRAFFIC = ENTITY_TRAIN | ENTITY_UNIT, ENTITY_ALL = ENTITY_TRAIN | ENTITY_UNIT | ENTITY_PARTIAL }

constants used for enumerating the contents of the database Functions

- _COMPILE_NMPB int NMPB08_LoadRailwayDatabase (const char *filename, bool log_stdout=false)
  loads the train database from an external file
- _COMPILE_NMPB int NMPB08_DumpRailwayDatabase (const char *filename=0)
  writes the contents of the database to a file
- _COMPILE_NMPB int NMPB08_EnumRailwayDatabase (EnumRailwayEntities enumProc, unsigned int include_types=ENTITY_TRAFFIC, void *userdata=0)
  enumerate the contents of the database
- _COMPILE_NMPB void * NMPB08_CreateRailwayTraffic (double nb_hours)
  create a context for converting railway traffic into an equivalent source model
- _COMPILE_NMPB int NMPB08_ClearRailwayTraffic (void *id)
  empties the traffic associated with the source model
- _COMPILE_NMPB int NMPB08_AddRailwayTraffic (void *id, const char *unit_or_train, double number, double speed)
  adds a traffic component to the source model
- _COMPILE_NMPB unsigned int NMPB08_SetRailwayOptions (void *id, unsigned int option, bool on_off)
  enables or disables selected options
- _COMPILE_NMPB unsigned int NMPB08_GetRailwayOptions (void *id, unsigned int option)
  return the currently selected options
- _COMPILE_NMPB unsigned int NMPB08_SetRailCorrection (void *id, NMPB_RailCorrectionType corr)
  set the rail correction
- _COMPILE_NMPB NMPB_RailCorrectionType NMPB08_GetRailCorrection (void *id)
  returns the current value of rail correction
- _COMPILE_NMPB RailwayEmission const * NMPB08_GetRailwayEmission (void *id, ScreenBodyInteraction *screenBodyInteraction=0)
  returns the equivalent source model for the current traffic state
- _COMPILE_NMPB int NMPB08_SetRailwayEmissionAngles (void *id, double sin_h, double sin_v)
  sets de emission angles for the equivalent source model
- _COMPILE_NMPB int NMPB08_DeleteRailwayTraffic (void *id)
  destroys the internal data structure used for converting railway traffic data into an equivalent source model.
- _COMPILE_NMPB int NMPB08_EnumRailwaySources (EnumRailwaySources enumProc, const char *name, void *userdata)
  enumerate the equivalent acoustical sources associated with a train or unit

Variables

- const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_BETON = 0.0
- const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_AUTRE = 3.0
- const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_BETON = 3.0
- const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_AUTRE = 6.0
- const NMPB_RailCorrectionType RAIL_ZONE_APPAREILS_VOIE = 6.0

Define Documentation

#define _COMPILE_NMPB extern "C"

compiler specific options for creating shared libraries

Definition at line 54 of file RailwayEmissionNMPB08.h.

Typedef Documentation

typedef bool(* EnumRailwayEntities)(RailwayEntity const &info, void *userdata)

user defined callback function for enumerating the contents of the database

Definition at line 269 of file RailwayEmissionNMPB08.h.

typedef bool(* EnumRailwaySources)(RailwayElementarySource const &info, void *userdata)

user defined callback function for enumerating the equivalent acoustical sources associated with a train or a unit

Definition at line 447 of file RailwayEmissionNMPB08.h.

typedef double NMPB_RailCorrectionType

correction for rail and mounting conditions as specified in the document "Production des cartes strategiques des grands axes routiers et ferrovaires", published by SETRA, August 2007.

Note that the following values are indicative and not integral part of the method.

- RAIL_LONG_TRAVERSES_BETON = 0 dB(A)
- RAIL_LONG_TRAVERSES_AUTRE = 3 dB(A)
- RAIL_COURT_TRAVERSES_BETON = 3 dB(A)
- RAIL_COURT_TRAVERSES_AUTRE = 6 dB(A)
-
- The end user may supply other values, e.g. for points and metal bridges

Definition at line 126 of file RailwayEmissionNMPB08.h.

Enumeration Type Documentation

enum NMPB_RailwayEmission_Entities

constants used for enumerating the contents of the database

Enumerator:

ENTITY_UNIT unit

ENTITY_TRAIN train

ENTITY_PARTIAL unit, only to be used as part of a train

ENTITY_TRAFFIC traffic units only

ENTITY_ALL all units and trains

ENTITY_UNIT unit

ENTITY_TRAIN train

ENTITY_PARTIAL unit, only to be used as part of a train

ENTITY_TRAFFIC traffic units only

ENTITY_ALL all units and trains

ENTITY_UNIT unit

ENTITY_TRAIN train

ENTITY_PARTIAL unit, only to be used as part of a train

ENTITY_TRAFFIC traffic units only

ENTITY_ALL all units and trains

Definition at line 242 of file RailwayEmissionNMPB08.h.

enum NMPB_RailwayEmission_HorizontalDirectivity

predefined horizontal directivity models

Enumerator:

HDIR_OMNI omnidirectionnel

HDIR_NMPB horizontal directivity as defined in NF S31-133

HDIR_OMNI omnidirectionnel

HDIR_NMPB horizontal directivity as defined in NF S31-133

HDIR_OMNI omnidirectionnel

HDIR_NMPB horizontal directivity as defined in NF S31-133

Definition at line 75 of file RailwayEmissionNMPB08.h.

enum NMPB_RailwayEmission_Options

options that modify the behaviour of the calculation model

Note:

OPTION_NO_DIRECTIVITY is useful if the directivity of the railway sources is integrated as part of the propagation model
OPTION_MODIFY_HEIGHTS is useful to adjust databases for different propagation models, i.e. when source heights in the database do not match those of the emission model

Enumerator:

OPTION_TRACE_DEBUG trace intermediate results to stdout

OPTION_NO_DIRECTIVITY do not include directivity

OPTION_MODIFY_HEIGHTS distribute sound powers over source heights

OPTION_DEYGOUT_DIFFRACTION use Deygout approximation for body/barrier interaction

OPTION_TRACE_DEBUG trace intermediate results to stdout

OPTION_NO_DIRECTIVITY do not include directivity

OPTION_MODIFY_HEIGHTS distribute sound powers over source heights

OPTION_DEYGOUT_DIFFRACTION use Deygout approximation for body/barrier interaction

OPTION_TRACE_DEBUG trace intermediate results to stdout

OPTION_NO_DIRECTIVITY do not include directivity

OPTION_MODIFY_HEIGHTS distribute sound powers over source heights

OPTION_DEYGOUT_DIFFRACTION use Deygout approximation for body/barrier interaction

Definition at line 103 of file RailwayEmissionNMPB08.h.

enum NMPB_RailwayEmission_VerticalDirectivity

predefined vertical directivity models

Enumerator:

VDIR_OMNI omnidirectional

VDIR_NMPB vertical directivity as defined in NF S31-133

VDIR_HEMI hemispheric emission upwards

VDIR_OMNI omnidirectional

VDIR_NMPB vertical directivity as defined in NF S31-133

VDIR_HEMI hemispheric emission upwards

VDIR_OMNI omnidirectional

VDIR_NMPB vertical directivity as defined in NF S31-133

VDIR_HEMI hemispheric emission upwards

Definition at line 85 of file RailwayEmissionNMPB08.h.

enum NPMP08_RailwayEmission_Error

error types returned from the NMPB08_Railway software library

Enumerator:

ERROR_XML_PARSER file could not be openened or parsed

ERROR_XML_SCHEMA incompatible XML file

ERROR_OPEN_FILE file could not be openened

ERROR_INVALID_ID identifier not defined in database

ERROR_INVALID_HEIGHT source height not supported in emission model

ERROR_XML_PARSER file could not be openened or parsed

ERROR_XML_SCHEMA incompatible XML file

ERROR_OPEN_FILE file could not be openened

ERROR_INVALID_ID identifier not defined in database

ERROR_INVALID_HEIGHT source height not supported in emission model

ERROR_XML_PARSER file could not be openened or parsed

ERROR_XML_SCHEMA incompatible XML file

ERROR_OPEN_FILE file could not be openened

ERROR_INVALID_ID identifier not defined in database

ERROR_INVALID_HEIGHT source height not supported in emission model

Definition at line 62 of file RailwayEmissionNMPB08.h.

Function Documentation

_COMPILE_NMPB int NMPB08_AddRailwayTraffic (void **id*, const char **unit_or_train*, double*number*, double*speed*)

adds a traffic component to the source model

Parameters:

id	handle to the internal structure used by the traffic model
unit_or_train	identification of the type of unit or train
number	number of units or trains passing during the reference period
speed	speed in km/h

Returns:

O if sucessful, an error code otherwise

Definition at line 681 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_ClearRailwayTraffic (void **id*)

empties the traffic associated with the source model

Parameters:

id	handle to the internal structure used by the traffic model

Returns:

O if sucessful, an error code otherwise

Definition at line 670 of file RailwayEmission.cpp.

_COMPILE_NMPB void* NMPB08_CreateRailwayTraffic (double*nb_hours*)

create a context for converting railway traffic into an equivalent source model

Parameters:

nb_hours	duration of the period of reference in hours

Returns:

a handle to an internal structure to be uses in consecutive calls to other function inside the library, a NULL pointer if error occured

Definition at line 659 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_DeleteRailwayTraffic (void **id*)

destroys the internal data structure used for converting railway traffic data into an equivalent source model.

Parameters:

id	handle to an internal data structure as returned by /ref NMPB08_CreateRailwayTraffic.

Returns:

0 if successful, an error code otherwise.

Definition at line 703 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_DumpRailwayDatabase (const char **filename* = `0`)

writes the contents of the database to a file

Parameters:

filename	the name of the output file in case the filename is a zero pointer or a zero length string, the output will be written to stdout

Returns:

0 if successful, otherwise an error code

Definition at line 16 of file RailwayDatabase.cpp.

_COMPILE_NMPB int NMPB08_EnumRailwayDatabase (EnumRailwayEntities*enumProc*, unsigned int*include_types* = `ENTITY_TRAFFIC`, void **userdata* = `0`)

enumerate the contents of the database

Parameters:

enumProc	user defined callback function
include_types	type of database entries to be enumerated
userdata	user defined data to be passed to the callback function

Definition at line 21 of file RailwayDatabase.cpp.

_COMPILE_NMPB int NMPB08_EnumRailwaySources (EnumRailwaySources *enumProc*, const char **name*, void **userdata*)

enumerate the equivalent acoustical sources associated with a train or unit

Parameters:

enumProc	user defined callback function
name	identification a train or unit
userdata	user defined data to be passed to the callback function

Definition at line 28 of file RailwayDatabase.cpp.

_COMPILE_NMPB NMPB_RailCorrectionType NMPB08_GetRailCorrection (void **id*)

returns the current value of rail correction

Parameters:

id	handle to the internal structure used by the traffic model

Returns:

the current value of the rail correction

_COMPILE_NMPB RailwayEmission const* NMPB08_GetRailwayEmission (void **id*, ScreenBodyInteraction **screenBodyInteraction* = 0)

returns the equivalent source model for the current traffic state

Parameters:

id	handle to the internal structure used by the traffic model
screenBodyInteraction	configuration of interaction between the train body and a nearby screen One interaction corresponds to a double reflection, once on the barrier, once on the car body. The equivalent reflection coefficient of the car bodies is encoded in the database and will modify the equivalent sound power of the image sources created through reflection. Absorption on the inner side of the barrier is also taken into account (see NF S 31-133, section 7.4.6).

Returns:

a pointer to the equivalent source model, a null pointer in case an invalid source height has been detected in the selected trains and units

Note:

if OPTION_H_INTERPOLATE is set, sources heights defined in the database will be automatically adapted to the 3 sources heights defined in the NMPB model. This is achieved by distributing the acoustical power over the predefined source heights proportional to the difference in height

Definition at line 692 of file RailwayEmission.cpp.

_COMPILE_NMPB unsigned int NMPB08_GetRailwayOptions (void *id, unsigned intoption)

return the currently selected options

Parameters:

id	handle to the internal structure used by the traffic model
option	options to be reported

Returns:

the selected set of options

Definition at line 726 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_LoadRailwayDatabase (const char *filename, boollog_stdout = false)

loads the train database from an external file

Parameters:

filename	name of the external file
log_stdout	if true, prints messages to stdout

Returns:

0 if successful, otherwise an error code

Definition at line 11 of file RailwayDatabase.cpp.

_COMPILE_NMPB unsigned int NMPB08_SetRailCorrection (void *id, NMPB_RailCorrectionTypecorr)

set the rail correction

Parameters:

id	handle to the internal structure used by the traffic model
corr	new correction for the rail / mounting condition

Returns:

O if sucessful, an error code otherwise

Definition at line 748 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_SetRailwayEmissionAngles (void *id, doublesin_h, doublesin_v)

sets de emission angles for the equivalent source model

Parameters:

id	handle to the internal structure used by the traffic model
sin_h	sinus of the angle of the propagation direction with the vertical plane perpendicular to the track
sin_v	sinus of the angle of the propagation direction with the horizontal plane containing the track

Returns:

0 if successful, an error code otherwise

Note:

if the OPTION_NO_DIRECIVITY is set, this function has no effect on the reported equivalent source model

Definition at line 737 of file RailwayEmission.cpp.

_COMPILE_NMPB unsigned int NMPB08_SetRailwayOptions (void *id, unsigned int option, bool on_off)

enables or disables selected options

Parameters:

id	handle to the internal structure used by the traffic model
option	options to be set or cleared
on_off	if true, options will be set, otherwise options will be cleared

Returns:

the modified set of options

Definition at line 715 of file RailwayEmission.cpp.

Variable Documentation

const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_AUTRE = 6.0

Definition at line 131 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_BETON = 3.0

Definition at line 130 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_AUTRE = 3.0

Definition at line 129 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_BETON = 0.0

Definition at line 128 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_ZONE_APPAREILS_VOIE = 6.0

Definition at line 132 of file RailwayEmissionNMPB08.h.

RailwayEmissionNMPB08.h File Reference

Classes

- struct RailwayEquivalentSource
- *elementary emission data represents apparent sound power for an equivalent line source emitted in a given direction* struct RailwayEmission
- *the emission associated with a railway traffic is represented as a set of elementary point sources with associated sound power and directivity* struct RailwayTrafficComponent
- *a single railway traffic component. Each component is defined as a number of units (or trains) of a given type that ciruclate at the same speed on the same track, the same speed. A complete traffic may be composed of one or more components.* struct RailwayTraffic
- struct ScreenBodyInteraction
- struct RailwayEntity
- *structure used for enumerating the contents of the database* struct RailwayElementarySource

structure used for enumerating the equivalent acoustical sources associated with a train or a unit Defines

- #define _COMPILE_NMPB extern "C"
  compiler specific options for creating shared libraries

Typedefs

- typedef double NMPB_RailCorrectionType
  correction for rail and mounting conditions as specified in the document "Production des cartes strategiques des grands axes routiers et ferrovaires", published by SETRA, August 2007.

  Note that the following values are indicative and not integral part of the method.

- typedef bool(* EnumRailwayEntities)(RailwayEntity const &info, void *userdata)
  user defined callback function for enumerating the contents of the database
- typedef bool(* EnumRailwaySources)(RailwayElementarySource const &info, void *userdata)
  user defined callback function for enumerating the equivalent acoustical sources associated with a train or a unit

Enumerations

- enum NPMP08_RailwayEmission_Error { ERROR_XML_PARSER = 1, ERROR_XML_SCHEMA = 2, ERROR_OPEN_FILE = 3, ERROR_INVALID_ID = 4, ERROR_INVALID_HEIGHT = 5, ERROR_XML_PARSER = 1, ERROR_XML_SCHEMA = 2, ERROR_OPEN_FILE = 3, ERROR_INVALID_ID = 4, ERROR_INVALID_HEIGHT = 5, ERROR_XML_PARSER = 1, ERROR_XML_SCHEMA = 2, ERROR_OPEN_FILE = 3, ERROR_INVALID_ID = 4, ERROR_INVALID_HEIGHT = 5 }
- *error types returned from the NMPB08_Railway software library* enum NMPB_RailwayEmission_HorizontalDirectivity { HDIR_OMNI = 0, HDIR_NMPB = 1, HDIR_OMNI = 0, HDIR_NMPB = 1, HDIR_OMNI = 0, HDIR_NMPB = 1 }
- *predefined horizontal directivity models* enum NMPB_RailwayEmission_VerticalDirectivity { VDIR_OMNI = 0, VDIR_NMPB = 1, VDIR_HEMI = 2, VDIR_OMNI = 0, VDIR_NMPB = 1, VDIR_HEMI = 2, VDIR_OMNI = 0, VDIR_NMPB = 1, VDIR_HEMI = 2 }

- *predefined vertical directivity models* enum NMPB_RailwayEmission_Options { OPTION_TRACE_DEBUG = 1, OPTION_NO_DIRECTIVITY = 2, OPTION_MODIFY_HEIGHTS = 4, OPTION_DEYGOUT_DIFFRACTION = 8, OPTION_TRACE_DEBUG = 1, OPTION_NO_DIRECTIVITY = 2, OPTION_MODIFY_HEIGHTS = 4, OPTION_DEYGOUT_DIFFRACTION = 8, OPTION_TRACE_DEBUG = 1, OPTION_NO_DIRECTIVITY = 2, OPTION_MODIFY_HEIGHTS = 4, OPTION_DEYGOUT_DIFFRACTION = 8 }
- *options that modify the behaviour of the calculation model* enum NMPB_RailwayEmission_Entities { ENTITY_UNIT = 1, ENTITY_TRAIN = 2, ENTITY_PARTIAL = 4, ENTITY_TRAFFIC = ENTITY_TRAIN | ENTITY_UNIT, ENTITY_ALL = ENTITY_TRAIN | ENTITY_UNIT | ENTITY_PARTIAL, ENTITY_UNIT = 1, ENTITY_TRAIN = 2, ENTITY_PARTIAL = 4, ENTITY_TRAFFIC = ENTITY_TRAIN | ENTITY_UNIT, ENTITY_ALL = ENTITY_TRAIN | ENTITY_UNIT | ENTITY_PARTIAL, ENTITY_UNIT = 1, ENTITY_TRAIN = 2, ENTITY_PARTIAL = 4, ENTITY_TRAFFIC = ENTITY_TRAIN | ENTITY_UNIT, ENTITY_ALL = ENTITY_TRAIN | ENTITY_UNIT | ENTITY_PARTIAL }

constants used for enumerating the contents of the database Functions

- _COMPILE_NMPB int NMPB08_LoadRailwayDatabase (const char *filename, bool log_stdout=false)
  loads the train database from an external file
- _COMPILE_NMPB int NMPB08_DumpRailwayDatabase (const char *filename=0)
  writes the contents of the database to a file
- _COMPILE_NMPB int NMPB08_EnumRailwayDatabase (EnumRailwayEntities enumProc, unsigned int include_types=ENTITY_TRAFFIC, void *userdata=0)
  enumerate the contents of the database
- _COMPILE_NMPB void * NMPB08_CreateRailwayTraffic (double nb_hours)
  create a context for converting railway traffic into an equivalent source model
- _COMPILE_NMPB int NMPB08_ClearRailwayTraffic (void *id)
  empties the traffic associated with the source model
- _COMPILE_NMPB int NMPB08_AddRailwayTraffic (void *id, const char *unit_or_train, double number, double speed)
  adds a traffic component to the source model
- _COMPILE_NMPB unsigned int NMPB08_SetRailwayOptions (void *id, unsigned int option, bool on_off)
  enables or disables selected options
- _COMPILE_NMPB unsigned int NMPB08_GetRailwayOptions (void *id, unsigned int option)
  return the currently selected options
- _COMPILE_NMPB unsigned int NMPB08_SetRailCorrection (void *id, NMPB_RailCorrectionType corr)
  set the rail correction
- _COMPILE_NMPB NMPB_RailCorrectionType NMPB08_GetRailCorrection (void *id)
  returns the current value of rail correction
- _COMPILE_NMPB RailwayEmission const * NMPB08_GetRailwayEmission (void *id, ScreenBodyInteraction *screenBodyInteraction=0)
  returns the equivalent source model for the current traffic state
- _COMPILE_NMPB int NMPB08_SetRailwayEmissionAngles (void *id, double sin_h, double sin_v)
  sets de emission angles for the equivalent source model
- _COMPILE_NMPB int NMPB08_DeleteRailwayTraffic (void *id)
  destroys the internal data structure used for converting railway traffic data into an equivalent source model.
- _COMPILE_NMPB int NMPB08_EnumRailwaySources (EnumRailwaySources enumProc, const char *name, void *userdata)
  enumerate the equivalent acoustical sources associated with a train or unit

Variables

- const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_BETON = 0.0
- const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_AUTRE = 3.0
- const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_BETON = 3.0
- const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_AUTRE = 6.0
- const NMPB_RailCorrectionType RAIL_ZONE_APPAREILS_VOIE = 6.0

Define Documentation

#define _COMPILE_NMPB extern "C"

compiler specific options for creating shared libraries

Definition at line 54 of file RailwayEmissionNMPB08.h.

Typedef Documentation

typedef bool(* EnumRailwayEntities)(RailwayEntity const &info, void *userdata)

user defined callback function for enumerating the contents of the database

Definition at line 269 of file RailwayEmissionNMPB08.h.

typedef bool(* EnumRailwaySources)(RailwayElementarySource const &info, void *userdata)

user defined callback function for enumerating the equivalent acoustical sources associated with a train or a unit

Definition at line 447 of file RailwayEmissionNMPB08.h.

typedef double NMPB_RailCorrectionType

correction for rail and mounting conditions as specified in the document "Production des cartes strategiques des grands axes routiers et ferrovaires", published by SETRA, August 2007.

Note that the following values are indicative and not integral part of the method.

- RAIL_LONG_TRAVERSES_BETON = 0 dB(A)
- RAIL_LONG_TRAVERSES_AUTRE = 3 dB(A)
- RAIL_COURT_TRAVERSES_BETON = 3 dB(A)
- RAIL_COURT_TRAVERSES_AUTRE = 6 dB(A)
-
- The end user may supply other values, e.g. for points and metal bridges

Definition at line 126 of file RailwayEmissionNMPB08.h.

Enumeration Type Documentation

enum NMPB_RailwayEmission_Entities

constants used for enumerating the contents of the database

Enumerator:

ENTITY_UNIT unit

ENTITY_TRAIN train

ENTITY_PARTIAL unit, only to be used as part of a train

ENTITY_TRAFFIC traffic units only

ENTITY_ALL all units and trains

ENTITY_UNIT unit

ENTITY_TRAIN train

ENTITY_PARTIAL unit, only to be used as part of a train

ENTITY_TRAFFIC traffic units only

ENTITY_ALL all units and trains

ENTITY_UNIT unit

ENTITY_TRAIN train

ENTITY_PARTIAL unit, only to be used as part of a train

ENTITY_TRAFFIC traffic units only

ENTITY_ALL all units and trains

Definition at line 242 of file RailwayEmissionNMPB08.h.

enum NMPB_RailwayEmission_HorizontalDirectivity

predefined horizontal directivity models

Enumerator:

HDIR_OMNI omnidirectionnel

HDIR_NMPB horizontal directivity as defined in NF S31-133

HDIR_OMNI omnidirectionnel

HDIR_NMPB horizontal directivity as defined in NF S31-133

HDIR_OMNI omnidirectionnel

HDIR_NMPB horizontal directivity as defined in NF S31-133

Definition at line 75 of file RailwayEmissionNMPB08.h.

enum NMPB_RailwayEmission_Options

options that modify the behaviour of the calculation model

Note:

OPTION_NO_DIRECTIVITY is useful if the directivity of the railway sources is integrated as part of the propagation model
OPTION_MODIFY_HEIGHTS is useful to adjust databases for different propagation models, i.e. when source heights in the database do not match those of the emission model

Enumerator:

OPTION_TRACE_DEBUG trace intermediate results to stdout

OPTION_NO_DIRECTIVITY do not include directivity

OPTION_MODIFY_HEIGHTS distribute sound powers over source heights

OPTION_DEYGOUT_DIFFRACTION use Deygout approximation for body/barrier interaction

OPTION_TRACE_DEBUG trace intermediate results to stdout

OPTION_NO_DIRECTIVITY do not include directivity

OPTION_MODIFY_HEIGHTS distribute sound powers over source heights

OPTION_DEYGOUT_DIFFRACTION use Deygout approximation for body/barrier interaction

OPTION_TRACE_DEBUG trace intermediate results to stdout

OPTION_NO_DIRECTIVITY do not include directivity

OPTION_MODIFY_HEIGHTS distribute sound powers over source heights

OPTION_DEYGOUT_DIFFRACTION use Deygout approximation for body/barrier interaction

Definition at line 103 of file RailwayEmissionNMPB08.h.

enum NMPB_RailwayEmission_VerticalDirectivity

predefined vertical directivity models

Enumerator:

VDIR_OMNI omnidirectional

VDIR_NMPB vertical directivity as defined in NF S31-133

VDIR_HEMI hemispheric emission upwards

VDIR_OMNI omnidirectional

VDIR_NMPB vertical directivity as defined in NF S31-133

VDIR_HEMI hemispheric emission upwards

VDIR_OMNI omnidirectional

VDIR_NMPB vertical directivity as defined in NF S31-133

VDIR_HEMI hemispheric emission upwards

Definition at line 85 of file RailwayEmissionNMPB08.h.

enum NPMP08_RailwayEmission_Error

error types returned from the NMPB08_Railway software library

Enumerator:

ERROR_XML_PARSER file could not be openened or parsed

ERROR_XML_SCHEMA incompatible XML file

ERROR_OPEN_FILE file could not be openened

ERROR_INVALID_ID identifier not defined in database

ERROR_INVALID_HEIGHT source height not supported in emission model

ERROR_XML_PARSER file could not be openened or parsed

ERROR_XML_SCHEMA incompatible XML file

ERROR_OPEN_FILE file could not be openened

ERROR_INVALID_ID identifier not defined in database

ERROR_INVALID_HEIGHT source height not supported in emission model

ERROR_XML_PARSER file could not be openened or parsed

ERROR_XML_SCHEMA incompatible XML file

ERROR_OPEN_FILE file could not be openened

ERROR_INVALID_ID identifier not defined in database

ERROR_INVALID_HEIGHT source height not supported in emission model

Definition at line 62 of file RailwayEmissionNMPB08.h.

Function Documentation

_COMPILE_NMPB int NMPB08_AddRailwayTraffic (void ***id*, const char **unit_or_train*, double*number*, double*speed***)

adds a traffic component to the source model

Parameters:

id	handle to the internal structure used by the traffic model
unit_or_train	identification of the type of unit or train
number	number of units or trains passing during the reference period
speed	speed in km/h

Returns:

O if sucessful, an error code otherwise

Definition at line 681 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_ClearRailwayTraffic (void ***id**)

empties the traffic associated with the source model

Parameters:

id	handle to the internal structure used by the traffic model

Returns:

O if sucessful, an error code otherwise

Definition at line 670 of file RailwayEmission.cpp.

_COMPILE_NMPB void* NMPB08_CreateRailwayTraffic (double*nb_hours*)

create a context for converting railway traffic into an equivalent source model

Parameters:

nb_hours	duration of the period of reference in hours

Returns:

a handle to an internal structure to be uses in consecutive calls to other function inside the library, a NULL pointer if error occured

Definition at line 659 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_DeleteRailwayTraffic (void **id*)

destroys the internal data structure used for converting railway traffic data into an equivalent source model.

Parameters:

id	handle to an internal data structure as returned by /ref NMPB08_CreateRailwayTraffic.

Returns:

0 if successful, an error code otherwise.

Definition at line 703 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_DumpRailwayDatabase (const char **filename* = `0`)

writes the contents of the database to a file

Parameters:

filename	the name of the output file in case the filename is a zero pointer or a zero length string, the output will be written to stdout

Returns:

0 if successful, otherwise an error code

Definition at line 16 of file RailwayDatabase.cpp.

_COMPILE_NMPB int NMPB08_EnumRailwayDatabase (EnumRailwayEntities*enumProc*, unsigned int*include_types* = `ENTITY_TRAFFIC`, void **userdata* = `0`)

enumerate the contents of the database

Parameters:

enumProc	user defined callback function
include_types	type of database entries to be enumerated
userdata	user defined data to be passed to the callback function

Definition at line 21 of file RailwayDatabase.cpp.

_COMPILE_NMPB int NMPB08_EnumRailwaySources (EnumRailwaySources *enumProc*, const char **name*, void **userdata*)

enumerate the equivalent acoustical sources associated with a train or unit

Parameters:

enumProc	user defined callback function
name	identification a train or unit
userdata	user defined data to be passed to the callback function

Definition at line 28 of file RailwayDatabase.cpp.

_COMPILE_NMPB NMPB_RailCorrectionType NMPB08_GetRailCorrection (void **id*)

returns the current value of rail correction

Parameters:

id	handle to the internal structure used by the traffic model

Returns:

the current value of the rail correction

_COMPILE_NMPB RailwayEmission const* NMPB08_GetRailwayEmission (void **id*, ScreenBodyInteraction **screenBodyInteraction* = 0)

returns the equivalent source model for the current traffic state

Parameters:

id	handle to the internal structure used by the traffic model
screenBodyInteraction	configuration of interaction between the train body and a nearby screen One interaction corresponds to a double reflection, once on the barrier, once on the car body. The equivalent reflection coefficient of the car bodies is encoded in the database and will modify the equivalent sound power of the image sources created through reflection. Absorption on the inner side of the barrier is also taken into account (see NF S 31-133, section 7.4.6).

Returns:

a pointer to the equivalent source model, a null pointer in case an invalid source height has been detected in the selected trains and units

Note:

if OPTION_H_INTERPOLATE is set, sources heights defined in the database will be automatically adapted to the 3 sources heights defined in the NMPB model. This is achieved by distributing the acoustical power over the predefined source heights proportional to the difference in height

Definition at line 692 of file RailwayEmission.cpp.

_COMPILE_NMPB unsigned int NMPB08_GetRailwayOptions (void *id, unsigned intoption)

return the currently selected options

Parameters:

id	handle to the internal structure used by the traffic model
option	options to be reported

Returns:

the selected set of options

Definition at line 726 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_LoadRailwayDatabase (const char *filename, boollog_stdout = false)

loads the train database from an external file

Parameters:

filename	name of the external file
log_stdout	if true, prints messages to stdout

Returns:

0 if successful, otherwise an error code

Definition at line 11 of file RailwayDatabase.cpp.

_COMPILE_NMPB unsigned int NMPB08_SetRailCorrection (void *id, NMPB_RailCorrectionTypecorr)

set the rail correction

Parameters:

id	handle to the internal structure used by the traffic model
corr	new correction for the rail / mounting condition

Returns:

O if sucessful, an error code otherwise

Definition at line 748 of file RailwayEmission.cpp.

_COMPILE_NMPB int NMPB08_SetRailwayEmissionAngles (void *id, doublesin_h, doublesin_v)

sets de emission angles for the equivalent source model

Parameters:

id	handle to the internal structure used by the traffic model
sin_h	sinus of the angle of the propagation direction with the vertical plane perpendicular to the track
sin_v	sinus of the angle of the propagation direction with the horizontal plane containing the track

Returns:

0 if successful, an error code otherwise

Note:

if the OPTION_NO_DIRECIVITY is set, this function has no effect on the reported equivalent source model

Definition at line 737 of file RailwayEmission.cpp.

_COMPILE_NMPB unsigned int NMPB08_SetRailwayOptions (void *id, unsigned int option, bool on_off)

enables or disables selected options

Parameters:

id	handle to the internal structure used by the traffic model
option	options to be set or cleared
on_off	if true, options will be set, otherwise options will be cleared

Returns:

the modified set of options

Definition at line 715 of file RailwayEmission.cpp.

Variable Documentation

const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_AUTRE = 6.0

Definition at line 131 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_COURT_TRAVERSES_BETON = 3.0

Definition at line 130 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_AUTRE = 3.0

Definition at line 129 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_LONG_TRAVERSES_BETON = 0.0

Definition at line 128 of file RailwayEmissionNMPB08.h.

const NMPB_RailCorrectionType RAIL_ZONE_APPAREILS_VOIE = 6.0

Definition at line 132 of file RailwayEmissionNMPB08.h.

RailwayDatabase.cpp File Reference

```
#include "../test_mem/safe_new.h"
#include "RailwayDatabase.h"
#include <math.h>
```

Defines

- #define Printf if (trace_debug) printf

Functions

- int NMPB08_LoadRailwayDatabase (const char *filename, bool trace)
  loads the train database from an external file
- int NMPB08_DumpRailwayDatabase (const char *filename)
  writes the contents of the database to a file
- int NMPB08_EnumRailwayDatabase (EnumRailwayEntities enumProc, unsigned int include_types, void *userdata)
  enumerate the contents of the database
- int NMPB08_EnumRailwaySources (EnumRailwaySources enumProc, const char *name, void *userdata)
  enumerate the equivalent acoustical sources associated with a train or unit
- template<class T > T MIN (T const &x, T const &y)
- template<class T > T MAX (T const &x, T const &y)
- std::string getID (const char *s)

Variables

- RailwayDatabase _shared_db

Define Documentation

#define Printf if (trace_debug) printf

Definition at line 38 of file RailwayDatabase.cpp.

Function Documentation

std::string getID (const char *s)

Definition at line 64 of file RailwayDatabase.cpp.

template<class T > T MAX (T const &*x*, T const &*y*) [inline]

Definition at line 57 of file RailwayDatabase.cpp.

template<class T > T MIN (T const &*x*, T const &*y*) [inline]

Definition at line 52 of file RailwayDatabase.cpp.

int NMPB08_DumpRailwayDatabase (const char *filename = 0)

writes the contents of the database to a file

Parameters:

filename	the name of the output file in case the filename is a zero pointer or a zero length string, the output will be written to stdout

Returns:

0 if successful, otherwise an error code

Definition at line 16 of file RailwayDatabase.cpp.

int NMPB08_EnumRailwayDatabase (EnumRailwayEntities enumProc, unsigned int include_types = ENTITY_TRAFFIC, void *userdata = 0)

enumerate the contents of the database

Parameters:

enumProc	user defined callback function
include_types	type of database entries to be enumerated
userdata	user defined data to be passed to the callback function

Definition at line 21 of file RailwayDatabase.cpp.

int NMPB08_EnumRailwaySources (EnumRailwaySources enumProc, const char *name, void *userdata)

enumerate the equivalent acoustical sources associated with a train or unit

Parameters:

enumProc	user defined callback function
name	identification a train or unit
userdata	user defined data to be passed to the callback function

Definition at line 28 of file RailwayDatabase.cpp.

int NMPB08_LoadRailwayDatabase (const char *filename, bool log_stdout = false)

loads the train database from an external file

Parameters:

filename	name of the external file
log_stdout	if true, prints messages to stdout

Returns:

0 if successful, otherwise an error code

Definition at line 11 of file RailwayDatabase.cpp.

Variable Documentation

RailwayDatabase _shared_db

Definition at line 5 of file RailwayDatabase.cpp.

RailwayDatabase.h File Reference

```
#include "RailwayEmissionNMPB08.h"
#include "../LectureXML/LectureXML.hpp"
#include <string>
#include <vector>
#include <map>
#include <assert.h>
```

Classes

- struct Spectrum
- struct RailwaySource
- *level 1 of the database source records describe elementary sources and their sound power as a function of operating conditions* struct RailwaySourcePosition
- *auxiliary structure for assigning and positionning an elementary source on a railway unit* struct RailwayUnit
- *level 2 of the database unit records describe individual rolling stock units* struct RailwayTrainUnit
- *auxiliary structure for assigning sequences of units to trains* struct RailwayTrain
- *level 3 of the database train records describe complete trains made up of one or more units* class RailwayDatabase

the database of railway rolling stock Defines

- #define Spectrum _Local_RWDBN8_Spectrum_
  storage for spectral data this simplified version assuming all data is stored as 18 third octave bands in the range 100 - 5000 Hz
- #define RailwaySource _Local_RWDBN8_RailwaySource_
- #define RailwaySourcePosition _Local_RWDBN8_SourcePosition_
- #define RailwayUnit _Local_RWDBN8_RailwayUnit_
- #define RailwayTrainUnit _Local_RWDBN8_RailwayTrainUnit_
- #define RailwayTrain _Local_RWDBN8_RailwayTrain_
- #define RailwayDatabase _Local_RWDBN8_RailwayDatabase_

Typedefs

- typedef std::string IDREF
  storage type for identifier/references in XML files

Define Documentation

#define RailwayDatabase _Local_RWDBN8_RailwayDatabase_

Definition at line 36 of file RailwayDatabase.h.

#define RailwaySource _Local_RWDBN8_RailwaySource_

Definition at line 31 of file RailwayDatabase.h.

#define RailwaySourcePosition _Local_RWDBN8_SourcePosition_

Definition at line 32 of file RailwayDatabase.h.

#define RailwayTrain _Local_RWDBN8_RailwayTrain_

Definition at line 35 of file RailwayDatabase.h.

#define RailwayTrainUnit _Local_RWDBN8_RailwayTrainUnit_

Definition at line 34 of file RailwayDatabase.h.

#define RailwayUnit _Local_RWDBN8_RailwayUnit_

Definition at line 33 of file RailwayDatabase.h.

#define Spectrum _Local_RWDBN8_Spectrum_

storage for spectral data this simplified version assuming all data is stored as 18 third octave bands in the range 100 - 5000 Hz

Definition at line 30 of file RailwayDatabase.h.

Typedef Documentation

IDREF

storage type for identifier/references in XML files

Definition at line 16 of file RailwayDatabase.h.

RailwayEmission.cpp File Reference

```
#include "RailwayEmission.h"
#include "stdlib.h"
#include <math.h>
```

Defines

- #define TRACE_DEBUG GetOptions(OPTION_TRACE_DEBUG)

Functions

- char * my_strdup (const char *s)
- template<class T > T MIN (T const &x, T const &y)
- template<class T > T MAX (T const &x, T const &y)
- std::string getID (const char *s)
- void * NMPB08_CreateRailwayTraffic (double nb_hours)
  create a context for converting railway traffic into an equivalent source model
- int NMPB08_ClearRailwayTraffic (void *id)
  empties the traffic associated with the source model
- int NMPB08_AddRailwayTraffic (void *id, const char *unit_or_train, double number, double speed)
  adds a traffic component to the source model
- RailwayEmission const * NMPB08_GetRailwayEmission (void *id, ScreenBodyInteraction *screenBodyInteraction)
  returns the equivalent source model for the current traffic state
- int NMPB08_DeleteRailwayTraffic (void *id)
  destroys the internal data structure used for converting railway traffic data into an equivalent source model.
- unsigned int NMPB08_SetRailwayOptions (void *id, unsigned int option, bool on_off)
  enables or disables selected options
- unsigned int NMPB08_GetRailwayOptions (void *id, unsigned int option)
  return the currently selected options
- int NMPB08_SetRailwayEmissionAngles (void *id, double sin_h, double sin_v)
  sets de emission angles for the equivalent source model
- unsigned int NMPB08_SetRailCorrection (void *id, NMPB_RailCorrectionType corr)
  set the rail correction
- NMPB_RailCorrectionType NMPB08_GetRailwayOptions (void *id)

Variables

- const double PI = 3.1415926

Define Documentation

#define TRACE_DEBUG GetOptions(OPTION_TRACE_DEBUG)

Definition at line 26 of file RailwayEmission.cpp.

Function Documentation

std::string getID (const char *s)

Definition at line 64 of file RailwayDatabase.cpp.

template<class T > T MAX (T const &x, T const &y) [inline]

Definition at line 52 of file RailwayEmission.cpp.

template<class T > T MIN (T const &x, T const &y) [inline]

Definition at line 47 of file RailwayEmission.cpp.

char* my_strdup (const char *s)

Definition at line 28 of file RailwayEmission.cpp.

int NMPB08_AddRailwayTraffic (void *id, const char *unit_or_train, double number, double speed)

adds a traffic component to the source model

Parameters:

id	handle to the internal structure used by the traffic model
unit_or_train	identification of the type of unit or train
number	number of units or trains passing during the reference period
speed	speed in km/h

Returns:

O if sucessful, an error code otherwise

Definition at line 681 of file RailwayEmission.cpp.

int NMPB08_ClearRailwayTraffic (void *id)

empties the traffic associated with the source model

Parameters:

id	handle to the internal structure used by the traffic model

Returns:

O if sucessful, an error code otherwise

Definition at line 670 of file RailwayEmission.cpp.

void* NMPB08_CreateRailwayTraffic (double nb_hours)

create a context for converting railway traffic into an equivalent source model

Parameters:

nb_hours	duration of the period of reference in hours

Returns:

a handle to an internal structure to be uses in consecutive calls to other function inside the library, a NULL pointer if error occured

Definition at line 659 of file RailwayEmission.cpp.

int NMPB08_DeleteRailwayTraffic (void *_id_)

destroys the internal data structure used for converting railway traffic data into an equivalent source model.

Parameters:

id	handle to an internal data structure as returned by /ref NMPB08_CreateRailwayTraffic.

Returns:

0 if successful, an error code otherwise.

Definition at line 703 of file RailwayEmission.cpp.

RailwayEmission const* NMPB08_GetRailwayEmission (void *_id_, ScreenBodyInteraction *_screenBodyInteraction_ = 0)

returns the equivalent source model for the current traffic state

Parameters:

id	handle to the internal structure used by the traffic model
screenBodyInteraction	configuration of interaction between the train body and a nearby screen One interaction corresponds to a double reflection, once on the barrier, once on the car body. The equivalent reflection coefficient of the car bodies is encoded in the database and will modify the equivalent sound power of the image sources created through reflection. Absorption on the inner side of the barrier is also taken into account (see NF S 31-133, section 7.4.6).

Returns:

a pointer to the equivalent source model, a null pointer in case an invalid source height has been detected in the selected trains and units

Note:

if OPTION_H_INTERPOLATE is set, sources heights defined in the database will be automatically adapted to the 3 sources heights defined in the NMPB model. This is achieved by distributing the acoustical power over the predefined source heights proportional to the difference in height

Definition at line 692 of file RailwayEmission.cpp.

unsigned int NMPB08_GetRailwayOptions (void *_id_, unsigned int_option_)

return the currently selected options

Parameters:

id	handle to the internal structure used by the traffic model
option	options to be reported

Returns:

the selected set of options

Definition at line 726 of file RailwayEmission.cpp.

NMPB_RailCorrectionType NMPB08_GetRailwayOptions (void *id)

Definition at line 759 of file RailwayEmission.cpp.

unsigned int NMPB08_SetRailCorrection (void *id, NMPB_RailCorrectionType corr)

set the rail correction

Parameters:

id	handle to the internal structure used by the traffic model
corr	new correction for the rail / mounting condition

Returns:

O if sucessful, an error code otherwise

Definition at line 748 of file RailwayEmission.cpp.

int NMPB08_SetRailwayEmissionAngles (void *id, double sin_h, double sin_v)

sets de emission angles for the equivalent source model

Parameters:

id	handle to the internal structure used by the traffic model
sin_h	sinus of the angle of the propagation direction with the vertical plane perpendicular to the track
sin_v	sinus of the angle of the propagation direction with the horizontal plane containing the track

Returns:

0 if successful, an error code otherwise

Note:

if the OPTION_NO_DIRECIVITY is set, this function has no effect on the reported equivalent source model

Definition at line 737 of file RailwayEmission.cpp.

unsigned int NMPB08_SetRailwayOptions (void *id, unsigned int option, bool on_off)

enables or disables selected options

Parameters:

id	handle to the internal structure used by the traffic model

option	options to be set or cleared
on_off	if true, options will be set, otherwise options will be cleared

Returns:

the modified set of options

Definition at line 715 of file RailwayEmission.cpp.

Variable Documentation

const double PI = 3.1415926

Definition at line 476 of file RailwayEmission.cpp.

RailwayEmission.h File Reference

```
#include "RailwayEmissionNMPB08.h"
#include "RailwayDatabase.h"
#include "math.h"
#include <vector>
```

Classes

- class RailwaySourceModel
- *the railway source model links traffic data to the sound power radiated from equivalent sources* class RailwaySourceModel::InvalidHeight
- *exception thrown when an invalid source is detected and OPTION_MODIFY_HEIGHTS is not set* struct **RailwaySourceModel::SourceDistribution**

distribution of sound power over different source heights Defines

- #define RailwaySourceModel _Local_RWEMN8_RailwaySourceModel_

Variables

- RailwayDatabase _shared_db

Define Documentation

#define RailwaySourceModel _Local_RWEMN8_RailwaySourceModel_

Definition at line 13 of file RailwayEmission.h.

Variable Documentation

RailwayDatabase _shared_db

Definition at line 5 of file RailwayDatabase.cpp.

RailwayEmissionNMPB.cpp File Reference

```
#include "stdafx.h"
```

Functions

- BOOL APIENTRY DllMain (HMODULE hModule, DWORD ul_reason_for_call, LPVOID lpReserved)

Function Documentation

BOOL APIENTRY DllMain (HMODULE *hModule*, DWORD *ul_reason_for_call*, LPVOID *lpReserved*)

Definition at line 11 of file RailwayEmissionNMPB.cpp.

stdafx.cpp File Reference

```
#include "stdafx.h"
```

stdafx.h File Reference

```
#include <windows.h>
```

Defines

- #define WINVER 0x0501
- #define _WIN32_WINNT 0x0501
- #define _WIN32_WINDOWS 0x0410
- #define _WIN32_IE 0x0600
- #define WIN32_LEAN_AND_MEAN

Define Documentation

#define _WIN32_IE 0x0600

Definition at line 23 of file stdafx.h.

#define _WIN32_WINDOWS 0x0410

Definition at line 19 of file stdafx.h.

#define _WIN32_WINNT 0x0501

Definition at line 15 of file stdafx.h.

#define WIN32_LEAN_AND_MEAN

Definition at line 26 of file stdafx.h.

#define WINVER 0x0501

Definition at line 11 of file stdafx.h.

Index

INDEX