TempusV2

Generated by Doxygen 1.7.4

Thu May 24 2012 17:55:14

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 2

Class Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

Namespace Documentation

4.1 Db Namespace Reference

Classes

- class Value
- class RowValue
- class Result
- class Connection

Functions

- template<>
  std::string **Value::as**< **std::string** > ()
- template<>
  Tempus::Time **Value::as**< **Tempus::Time** > ()

4.1.1 Detailed Description

Database access is modeled by means of the following classes, inspired by pqxx: A Db::Connection objet represents a connection to a database. It is a lightweighted objet that is reference-counted and thus can be copied safely. A Db::Result objet represents result of a query. It is a lightweighted objet that is reference-counted and thus can be copied safely. A Db::RowValue object represents a row of a result and is obtained by Db::Result::operator[] A Db::Value object represent a basic value. It is obtained by Db::RowValue::operator[]. It has templated conversion operators for common data types.

These classes throw std::runtime_error on problem.

4.2 Tempus Namespace Reference

Namespaces

- namespace Multimodal
- namespace PublicTransport
- namespace Road

Classes

- class Application
- struct ConsistentClass
- struct Base
- struct Time
- struct RoadType
- struct Point2D
- struct TransportType
- class ProgressionCallback
- struct TextProgression
- struct vertex_or_edge
- struct vertex_or_edge< G, boost::vertex_property_tag >
- struct vertex_or_edge< G, boost::edge_property_tag >
- struct FieldPropertyAccessor
- struct FunctionPropertyAccessor
- class PQImporter
- class Plugin
- class PluginGraphVisitorHelper
- class Request
- struct POI
- class Roadmap
- class MultiPlugin
- class LengthCalculator
- class PtPlugin
- class RoadPlugin
- class TestPlugin

Typedefs

- typedef unsigned long long int db_id_t
- typedef boost::gregorian::date Date
- typedef boost::posix_time::ptime DateTime
- typedef std::map< db_id_t, RoadType > RoadTypes
- typedef std::map< db_id_t, TransportType > TransportTypes
- typedef std::map< int, double > Costs

- typedef PluginGraphVisitorHelper< Road::Graph,&Plugin::road_vertex_accessor,&Plugin::road_edge_accessor > **PluginRoadGraphVisitor**
- typedef PluginGraphVisitorHelper< PublicTransport::Graph,&Plugin::pt_vertex_accessor,&Plugin::pt_edge_accessor > **PluginPtGraphVisitor**
- typedef PluginGraphVisitorHelper< Multimodal::Graph,&Plugin::vertex_accessor,&Plugin::edge_accessor > **PluginGraphVisitor**
- typedef std::list< Roadmap > Result

Enumerations

- enum CostId {
  CostDistance = 1, **CostDuration**, **CostPrice**, **CostCarbon**,
  CostCalories, **CostNumberOfChanges** }

Functions

- Point2D coordinates (const Road::Vertex &v, Db::Connection &db, const Road::Graph &graph)
- Point2D coordinates (const PublicTransport::Vertex &v, Db::Connection &db, const PublicTransport::Graph &graph)
- Point2D coordinates (const POI *poi, Db::Connection &db)
- Point2D coordinates (const Multimodal::Vertex &v, Db::Connection &db, const Multimodal::Graph &graph)
- ostream & **operator**<< (ostream &out, const Multimodal::Vertex &v)
- ostream & **operator**<< (ostream &out, const Multimodal::Edge &e)
- std::ostream & **operator**<< (std::ostream &out, const Multimodal::Vertex &v)
- std::ostream & **operator**<< (std::ostream &out, const Multimodal::Edge &v)
- template<class G >
  boost::graph_traits< G >::vertex_descriptor **vertex_from_id** (Tempus::db_id_t db_id, G &graph)
- template<class G >
  boost::graph_traits< G >::edge_descriptor **edge_from_id** (Tempus::db_id_t db_id, G &graph)
- template<class G >
  bool **vertex_exists** (typename boost::graph_traits< G >::vertex_descriptor v, G &graph)
- template<class G >
  bool **edge_exists** (typename boost::graph_traits< G >::edge_descriptor v, G &graph)
- **DECLARE_TEMPUS_PLUGIN** (MultiPlugin)
- **DECLARE_TEMPUS_PLUGIN** (PtPlugin)
- **DECLARE_TEMPUS_PLUGIN** (RoadPlugin)

Variables

- ProgressionCallback **null_progression_callback**

4.2.1 Detailed Description

Tempus PostgreSQL importer.

TestPlugin used for unit tests

4.2.2 Typedef Documentation

4.2.2.1 typedef std::map<int, double> Tempus::Costs

Type used to model costs. Either in a Step or as an optimizing criterion. This is a map to a double value and thus is user extensible.

4.2.2.2 typedef boost::gregorian::date Tempus::Date

Date type : dd/mm/yyyy

4.2.2.3 typedef boost::posix_time::ptime Tempus::DateTime

DateTime stores a date and a time

4.2.2.4 typedef unsigned long long int Tempus::db_id_t

Type used inside the DB to store IDs. O means NULL.

4.2.2.5 typedef std::list<Roadmap> Tempus::Result

A Result is a list of Roadmap, ordered by relevance towards optimizing criteria

4.2.2.6 typedef std::map<db_id_t, RoadType> Tempus::RoadTypes

Road types constants.

4.2.2.7 typedef std::map<db_id_t, TransportType> Tempus::TransportTypes

Transport types constants.

4.2.3 Enumeration Type Documentation

4.2.3.1 enum Tempus::CostId

Default common cost identifiers

4.2.4 Function Documentation

4.2.4.1 Point2D Tempus::coordinates (const Road::Vertex & *v,* Db::Connection & *db,* const Road::Graph & *graph*)

Get 2D coordinates of a road vertex, from the database

4.2.4.2 Point2D Tempus::coordinates (const PublicTransport::Vertex & *v,* Db::Connection & *db,* const PublicTransport::Graph & *graph*)

Get 2D coordinates of a public transport vertex, from the database

4.2.4.3 Point2D Tempus::coordinates (const Multimodal::Vertex & *v,* Db::Connection & *db,* const Multimodal::Graph & *graph*)

Get 2D coordinates of a multimodal vertex, from the database

4.2.4.4 Point2D Tempus::coordinates (const POI * *poi,* Db::Connection & *db*)

Get 2D coordinates of a POI, from the database

4.3 Tempus::Multimodal Namespace Reference

Classes

- struct Vertex
- struct Edge
- struct Graph
- struct VertexIterator
- struct OutEdgeIterator
- struct EdgeIterator
- class VertexIndexProperty
- class EdgeIndexProperty

Functions

- VertexIndexProperty get (boost::vertex_index_t, const Multimodal::Graph &graph)
- EdgeIndexProperty **get** (boost::edge_index_t, const Multimodal::Graph &graph)
- size_t **get** (const VertexIndexProperty &p, const Multimodal::Vertex &v)
- size_t **get** (const EdgeIndexProperty &p, const Multimodal::Edge &e)
- size_t num_vertices (const Graph &graph)
- size_t num_edges (const Graph &graph)
- Vertex & source (Edge &e, const Graph &graph)

- Vertex & target (Edge &e, const Graph &graph)
- pair< VertexIterator, VertexIterator > vertices (const Graph &graph)
- pair< EdgeIterator, EdgeIterator > edges (const Graph &graph)
- pair< OutEdgeIterator, OutEdgeIterator > out_edges (const Vertex &v, const Graph &graph)
- size_t **out_degree** (Vertex &v, const Graph &graph)
- std::pair< Edge, bool > edge (const Vertex &u, const Vertex &v, const Graph &graph)

4.3.1 Detailed Description

Multimodal namespace

A Multimodal::Graph is a Road::Graph and a list of PublicTransport::Graph

4.3.2 Function Documentation

4.3.2.1 std::pair< Edge, bool > Tempus::Multimodal::edge (const Vertex & *u*, const Vertex & *v*, const Graph & *graph*)

Find an edge, based on a source and target vertex. It does not implements Adjacency-Matrix, since it does not returns in constant time (linear in the number of edges)

4.3.2.2 std::pair< EdgeIterator, EdgeIterator > Tempus::Multimodal::edges (const Graph & *graph*)

Returns a range of EdgeIterator. Constant time

4.3.2.3 VertexIndexProperty Tempus::Multimodal::get (boost::vertex_index_t , const Multimodal::Graph & *graph*)

Overloading of get()

4.3.2.4 size_t Tempus::Multimodal::num_edges (const Graph & *graph*)

Number of edges. Constant time

4.3.2.5 size_t Tempus::Multimodal::num_vertices (const Graph & *graph*)

Number of vertices. Constant time

4.3.2.6 std::pair< OutEdgeIterator, OutEdgeIterator > Tempus::Multimodal::out_edges (const Vertex & *v*, const Graph & *graph*)

Returns a range of EdgeIterator that allows to iterate on out edges of a vertex. Constant time

4.3.2.7 Vertex & Tempus::Multimodal::source (Edge & *e*, const Graph & *graph*)

Returns source vertex from an edge. Constant time (linear in number of PT networks)

4.3.2.8 Vertex & Tempus::Multimodal::target (Edge & *e*, const Graph & *graph*)

Returns source vertex from an edge. Constant time (linear in number of PT networks)

4.3.2.9 std::pair< VertexIterator, VertexIterator > Tempus::Multimodal::vertices (const Graph & *graph*)

Returns a range of VertexIterator. Constant time

4.4 Tempus::PublicTransport Namespace Reference

Classes

- struct Network
- struct Stop
- struct Section
- struct Calendar
- struct Trip
- struct Route
- struct FareRule
- struct FareAttribute
- struct Transfer

Typedefs

- typedef boost::vecS VertexListType
- typedef boost::vecS **EdgeListType**
- typedef boost::mpl::if_< boost::detail::is_random_access< VertexListType >::type, size_t, void * >::type Vertex
- typedef boost::detail::edge_desc_impl< boost::directed_tag, Vertex > Edge

  see adjacency_list.hpp
- typedef boost::adjacency_list< VertexListType, EdgeListType, boost::directedS, Stop, Section > Graph

- typedef boost::graph_traits< Graph >::vertex_iterator **VertexIterator**
- typedef boost::graph_traits< Graph >::edge_iterator **EdgeIterator**
- typedef boost::graph_traits< Graph >::out_edge_iterator **OutEdgeIterator**

4.4.1 Detailed Description

A PublicTransport::Graph is a made of PublicTransport::Stop and PublicTransport::Section

It generally maps to the database's schema: one class exists for each table. Tables with 1<->N arity are represented by STL containers (vectors or lists) External keys are represented by pointers to other classes or by vertex/edge descriptors.

PublicTransport::Stop and PublicTransport::Section classes are used to build a BGL public transport graph.

4.4.2 Typedef Documentation

4.4.2.1 typedef boost::adjacency_list<VertexListType, EdgeListType, boost::directedS, Stop, Section> Tempus::PublicTransport::Graph

Definition of a public transport graph

4.4.2.2 typedef boost::mpl::if_<boost::detail::is_random_access<VertexListType>::type, size_t, void*>::type Tempus::PublicTransport::Vertex

To make a long line short: VertexDescriptor is either typedef'd to size_t or to a pointer, depending on VertexListType and EdgeListType used to represent lists of vertices (vecS, listS, etc.)

4.4.2.3 typedef boost::vecS Tempus::PublicTransport::VertexListType

storage types used to make a road graph

4.5 Tempus::Road Namespace Reference

Classes

- struct Node
- struct Section
- struct Road

Typedefs

- typedef boost::vecS VertexListType

- typedef boost::vecS **EdgeListType**
- typedef boost::mpl::if_< boost::detail::is_random_access< VertexListType >::type, size_t, void * >::type Vertex
- typedef boost::detail::edge_desc_impl< boost::undirected_tag, Vertex > Edge

  see adjacency_list.hpp
- typedef boost::adjacency_list< VertexListType, EdgeListType, boost::undirectedS, Node, Section > Graph
- typedef boost::graph_traits< Graph >::vertex_iterator **VertexIterator**
- typedef boost::graph_traits< Graph >::edge_iterator **EdgeIterator**
- typedef boost::graph_traits< Graph >::out_edge_iterator **OutEdgeIterator**

4.5.1 Detailed Description

A Road::Graph is made of Road::Node and Road::Section

It generally maps to the database's schema: one class exists for each table. Tables with 1<->N arity are represented by STL containers (vectors or lists) External keys are represented by reference to other classes or by vertex/edge descriptors

Road::Node and Road::Section classes are used to build a BGL road graph as "bundled" edge and vertex properties

4.5.2 Typedef Documentation

4.5.2.1 typedef boost::adjacency_list<VertexListType, EdgeListType, boost::undirectedS, Node, Section > Tempus::Road::Graph

The final road graph type

4.5.2.2 typedef boost::mpl::if_<boost::detail::is_random_access<VertexListType>::type, size_t, void*>::type Tempus::Road::Vertex

To make a long line short: VertexDescriptor is either typedef'd to size_t or to a pointer, depending on VertexListType and EdgeListType used to represent lists of vertices (vecS, listS, etc.)

4.5.2.3 typedef boost::vecS Tempus::Road::VertexListType

Storage types used to make a road graph

4.6 WPS Namespace Reference

Classes

- class StateService

- class ConnectService
- class PreBuildService
- class BuildService
- class PluginListService
- class ConstantListService
- class PluginService
- class GetOptionsDescService
- class GetMetricsService
- class GetOptionsService
- class SetOptionsService
- class PreProcessService
- class ProcessService
- class ResultService
- class CleanupService
- class Request
- class Service

Functions

- void **ensure_minimum_state** (int state)
- void **get_xml_point** (xmlNode *node, double &x, double &y)
- Tempus::db_id_t **road_vertex_id_from_coordinates** (Db::Connection &db, double x, double y)

Variables

- static StateService **state_service_**
- static GetMetricsService **get_metrics_service**
- static GetOptionsService **get_options_service**
- static GetOptionsDescService **get_option_desc_service**
- static SetOptionsService **set_option_service**
- static ConnectService **connect_service_**
- static PluginListService **plugin_list_service**
- static PreBuildService **pre_build_service_**
- static BuildService **build_service_**
- static PreProcessService **pre_process_service_**
- static ProcessService **process_service_**
- static ResultService **result_service_**
- static CleanupService **cleanup_service_**
- static ConstantListService **constant_list_service**

4.6.1 Detailed Description

A WPS Service is a generic process callable through the 'Execute' WPS operation.

4.7 wps_client Namespace Reference

Classes

- class HttpCgiConnection
- class WPSClient

Functions

- def **to_xml_indent**
- def **to_xml**
- def **get_wps_exception**

4.7.1 Detailed Description

```
WPS client classes.
Shared by the command line tests and by the QGis plugin
```

Chapter 5

Class Documentation

5.1 Tempus::Application Class Reference

Public Types

- enum State { Started = 0, Connected, GraphPreBuilt, GraphBuilt }

Public Member Functions

- void **state** (State state)
- State **state** () const
- void **connect** (const std::string &db_options)
- Db::Connection & **db_connection** ()
- const std::string & **db_options** () const
- Plugin * **load_plugin** (const std::string &name)
- void **unload_plugin** (Plugin *plugin)
- void **pre_build_graph** ()
- void **build_graph** ()
- Multimodal::Graph & **graph** ()

Static Public Member Functions

- static Application * **instance** ()

Protected Attributes

- Db::Connection **db_**
- std::string **db_options_**
- Multimodal::Graph **graph_**
- State **state_**

5.1.1 Member Enumeration Documentation

5.1.1.1 enum Tempus::Application::State

Used to represent the application state

Enumerator:

Started The application has just been (re)started.

Connected The application is connected to a database.

GraphPreBuilt Graph has been pre built.

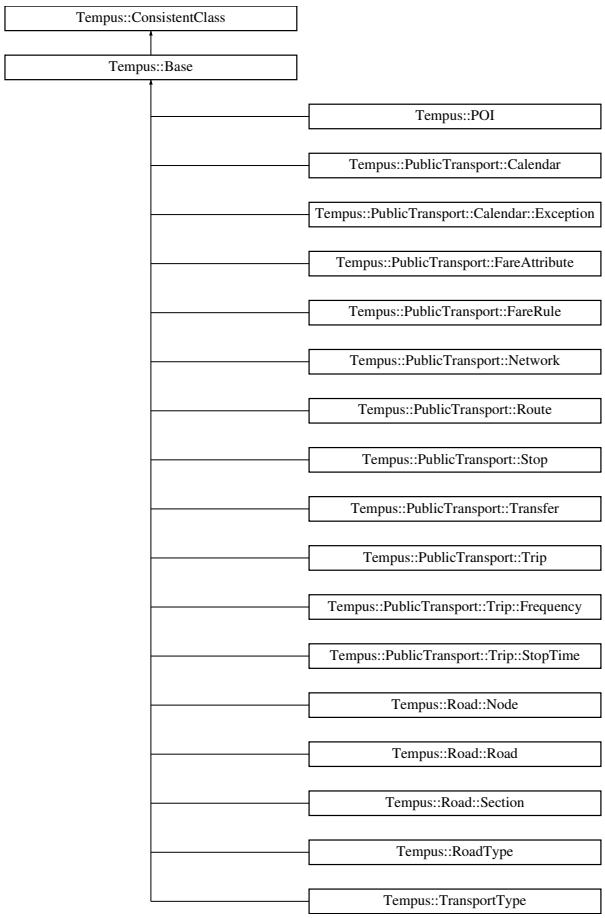
GraphBuilt Graph has been built.

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

- core/application.hh
- core/application.cc

5.2 Tempus::Base Struct Reference

Inheritance diagram for Tempus::Base:



Public Attributes

- db_id_t db_id

5.2.1 Member Data Documentation

5.2.1.1 db_id_t Tempus::Base::db_id

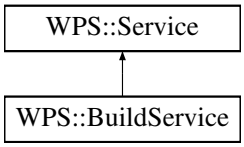
Persistant ID relative to the storage database. Common to many classes.

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

- core/common.hh

5.3 WPS::BuildService Class Reference

Inheritance diagram for WPS::BuildService:



Public Member Functions

- Service::ParameterMap & **execute** (Service::ParameterMap &input_parameter_map)

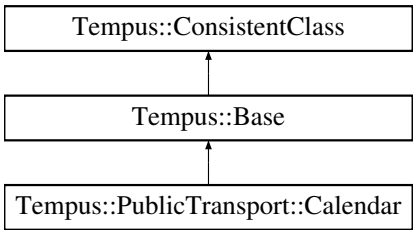
The documentation for this class was generated from the following file:

- wps/tempus_services.cc

5.4 Tempus::PublicTransport::Calendar Struct Reference

`#include <public_transport_graph.hh>`

Inheritance diagram for Tempus::PublicTransport::Calendar:



Classes

- struct Exception

Public Attributes

- bool **monday**
- bool **tuesday**
- bool **wednesday**
- bool **thursday**

- bool **friday**
- bool **saturday**
- bool **sunday**
- Date **start_date**
- Date **end_date**
- std::vector< Exception > **service_exceptions**

5.4.1 Detailed Description

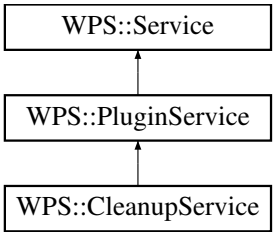
Refers to the 'pt_calendar' table

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

- core/public_transport_graph.hh

5.5 WPS::CleanupService Class Reference

Inheritance diagram for WPS::CleanupService:



Public Member Functions

- Service::ParameterMap & **execute** (ParameterMap &input_parameter_map)

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

- wps/tempus_services.cc

5.6 Db::Connection Class Reference

`#include <db.hh>`

Public Member Functions

- **Connection** (const std::string &db_options)

- void **connect** (const std::string &db_options)
- **Connection** (const Connection &r)
- Connection & **operator=** (const Connection &r)
- Result exec (const std::string &query) throw (std::runtime_error)

Protected Member Functions

- void **dec_refs** () const
- void **inc_refs** () const

Protected Attributes

- PGconn * **conn_**
- int **nrefs_**

5.6.1 Detailed Description

Class representing connection to a database.

5.6.2 Member Function Documentation

5.6.2.1 Result Db::Connection::exec (const std::string & *query*) throw (std::runtime_error) [inline]

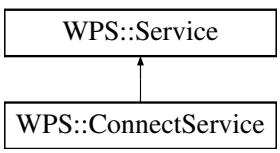
Query execution. Returns a Db::Result. Throws a std::runtime_error on problem

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

- core/db.hh

5.7 WPS::ConnectService Class Reference

Inheritance diagram for WPS::ConnectService:



Public Member Functions

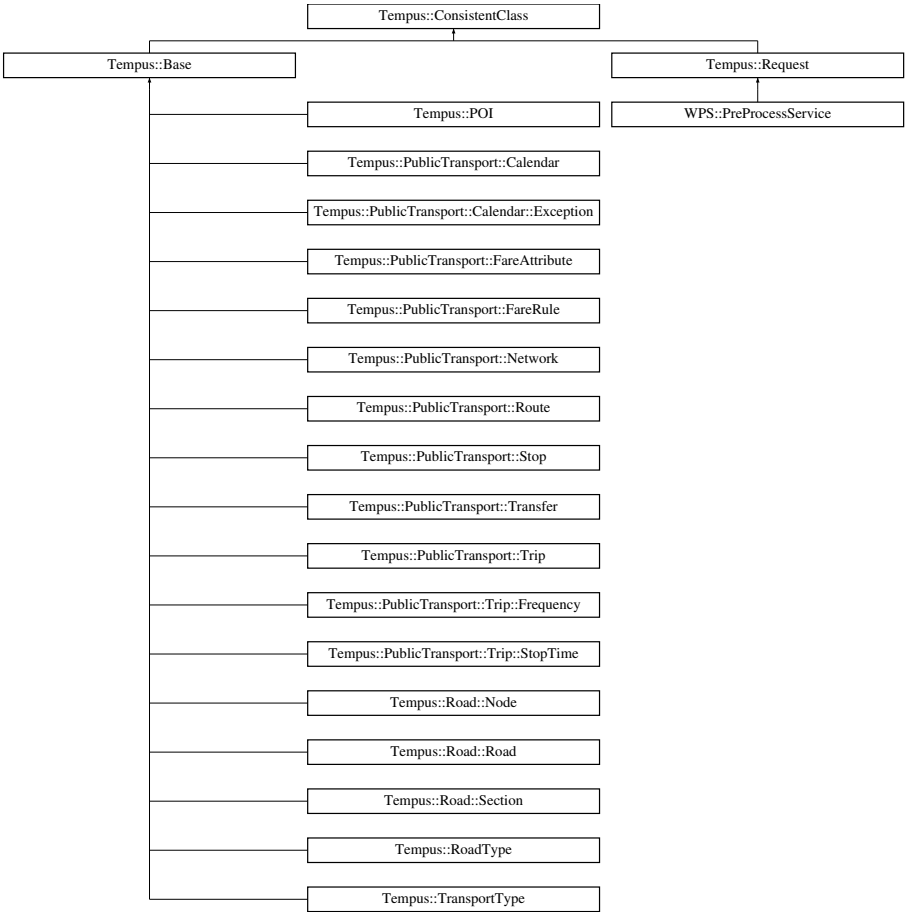
- Service::ParameterMap & **execute** (Service::ParameterMap &input_parameter_-map)

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

- wps/tempus_services.cc

5.8 Tempus::ConsistentClass Struct Reference

Inheritance diagram for Tempus::ConsistentClass:



Public Member Functions

- bool check_consistency ()

Protected Member Functions

- virtual bool check_consistency_ ()

5.8.1 Member Function Documentation

5.8.1.1 bool Tempus::ConsistentClass::check_consistency ( ) [inline]

Consistency checking. When on debug mode, calls the virtual check() method. When the debug mode is disabled, it does nothing.

5.8.1.2 virtual bool Tempus::ConsistentClass::check_consistency_ ( ) [inline, protected, virtual]

Private method to override in derived classes. Does nothing by default.

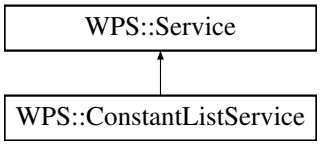
Reimplemented in Tempus::TransportType, Tempus::PublicTransport::Trip, Tempus::PublicTransport::Route, Tempus::PublicTransport::FareAttribute, Tempus::PublicTransport::Transfer, and Tempus::Request.

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

- core/common.hh

5.9 WPS::ConstantListService Class Reference

Inheritance diagram for WPS::ConstantListService:



Public Member Functions

- Service::ParameterMap & **execute** (ParameterMap &input_parameter_map)

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

- wps/tempus_services.cc

5.10 Tempus::Multimodal::Edge Struct Reference

`#include <multimodal_graph.hh>`

Public Types

- enum **ConnectionType** {
  UnknownConnection, **Road2Road**, **Road2Transport**, **Transport2Road**,
  Transport2Transport, **Road2Poi**, **Poi2Road** }

Public Member Functions

- ConnectionType **connection_type** () const
- **Edge** (Multimodal::Vertex s, Multimodal::Vertex t)
- bool **operator==** (const Multimodal::Edge &e) const
- bool **operator!=** (const Multimodal::Edge &e) const
- bool **operator**< (const Multimodal::Edge &e) const

Public Attributes

- Multimodal::Vertex **source**
- Multimodal::Vertex **target**

5.10.1 Detailed Description

A multimodal edge is a pair of multimodal vertices

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

- core/multimodal_graph.hh
- core/multimodal_graph.cc

5.11 Tempus::Multimodal::EdgeIndexProperty Class Reference

Public Member Functions

- **EdgeIndexProperty** (const Graph &graph)
- size_t **get_index** (const Edge &v) const
- size_t **operator[]** (const Edge &e) const

Protected Attributes

- const Multimodal::Graph & **graph_**

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

- core/multimodal_graph.hh

5.12 Tempus::Multimodal::EdgeIterator Struct Reference

Public Member Functions

- **EdgeIterator** (const Multimodal::Graph &graph)
- void **to_end** ()
- Multimodal::Edge & **dereference** () const
- void **increment** ()
- bool **equal** (const EdgeIterator &v) const

Protected Attributes

- const Multimodal::Graph * **graph_**
- Multimodal::VertexIterator **vi_**
- Multimodal::VertexIterator **vi_end_**
- Multimodal::OutEdgeIterator **ei_**
- Multimodal::OutEdgeIterator **ei_end_**

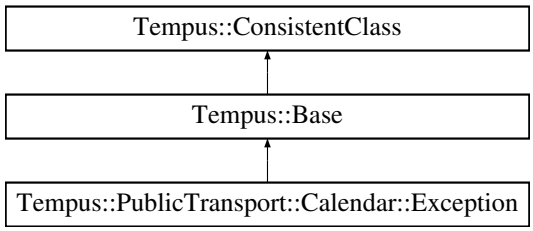
The documentation for this struct was generated from the following files:

- core/multimodal_graph.hh
- core/multimodal_graph.cc

5.13 Tempus::PublicTransport::Calendar::Exception Struct Reference

`#include <public_transport_graph.hh>`

Inheritance diagram for Tempus::PublicTransport::Calendar::Exception:



Public Types

- enum **ExceptionType** { **ServiceAdded** = 1, **ServiceRemoved** }

Public Attributes

- Date **calendar_date**
- ExceptionType **exception_type**

5.13.1 Detailed Description

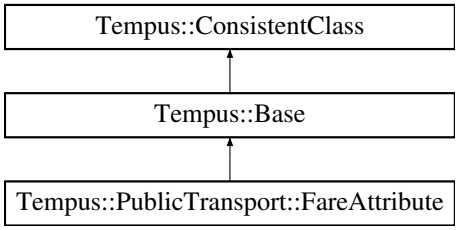
Refers to the 'pt_calendar_date' table. It represents exceptions to the regular service

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

- core/public_transport_graph.hh

5.14 Tempus::PublicTransport::FareAttribute Struct Reference

Inheritance diagram for Tempus::PublicTransport::FareAttribute:



Public Types

- enum **TransferType** { **NoTransferAllowed** = 0, **OneTransferAllowed**, **TwoTransfersAllowed**, **UnlimitedTransfers** = -1 }
- typedef std::vector< FareRule > **FareRulesList**

Public Attributes

- char currency_type [4]

  ISO 4217 codes.
- double **price**
- int **transfers**
- int transfers_duration

  in seconds
- FareRulesList **fare_rules**

Protected Member Functions

- bool check_consistency_ ()
- FareAttribute ()

5.14.1 Constructor & Destructor Documentation

5.14.1.1 Tempus::PublicTransport::FareAttribute::FareAttribute ( ) [inline, protected]

< default value

5.14.2 Member Function Documentation

5.14.2.1 bool Tempus::PublicTransport::FareAttribute::check_consistency_ ( ) [inline, protected, virtual]

Private method to override in derived classes. Does nothing by default.

Reimplemented from Tempus::ConsistentClass.

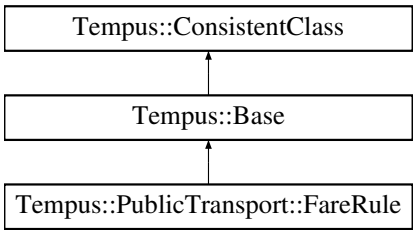
The documentation for this struct was generated from the following file:

- core/public_transport_graph.hh

5.15 Tempus::PublicTransport::FareRule Struct Reference

`#include <public_transport_graph.hh>`

Inheritance diagram for Tempus::PublicTransport::FareRule:



Public Types

- typedef std::vector< int > **ZoneIdList**

Public Attributes

- Route * **route**
- ZoneIdList **origins**
- ZoneIdList **destinations**
- ZoneIdList **contains**

5.15.1 Detailed Description

Refers to the 'pt_fare_rule' table

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

- core/public_transport_graph.hh

5.16 Tempus::FieldPropertyAccessor< Graph, Tag, T, Member > Struct Template Reference

`#include <multimodal_graph.hh>`

Public Member Functions

- **FieldPropertyAccessor** (Graph &graph, Member mem)

Public Attributes

- Graph & **graph_**
- Member **mem_**

5.16.1 Detailed Description

template<class Graph, class Tag, class T, class Member>struct Tempus::FieldPropertyAccessor< Graph, Tag, T, Member >

A FieldPropertyAccessor implements a Readable Property Map concept and gives read access to the member of a vertex or edge

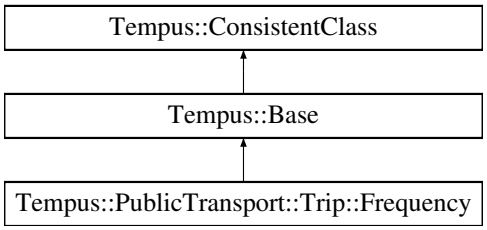
The documentation for this struct was generated from the following file:

- core/multimodal_graph.hh

5.17 Tempus::PublicTransport::Trip::Frequency Struct Reference

`#include <public_transport_graph.hh>`

Inheritance diagram for Tempus::PublicTransport::Trip::Frequency:



Public Attributes

- Time **start_time**
- Time **end_time**
- int **headways_secs**

5.17.1 Detailed Description

Refers to the 'pt_frequency' table

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

- core/public_transport_graph.hh

5.18 Tempus::FunctionPropertyAccessor< Graph, Tag, T, Function > Struct Template Reference

`#include <multimodal_graph.hh>`

Public Member Functions

- **FunctionPropertyAccessor** (Graph &graph, Function fct)

Public Attributes

- Graph & **graph_**
- Function **fct_**

5.18.1 Detailed Description

template<class Graph, class Tag, class T, class Function>struct Tempus::FunctionPropertyAccessor< Graph, Tag, T, Function >

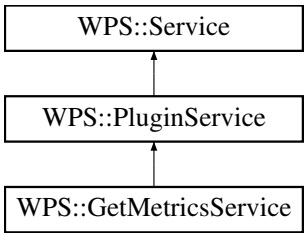
A FunctionPropertyAccessor implements a Readable Property Map concept by means of a function application on a vertex or edge of a graph

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

- core/multimodal_graph.hh

5.19 WPS::GetMetricsService Class Reference

Inheritance diagram for WPS::GetMetricsService:



Public Member Functions

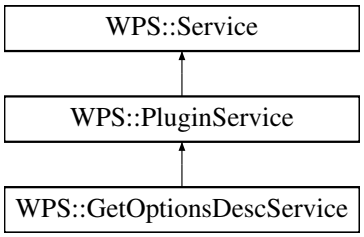
- Service::ParameterMap & **execute** (ParameterMap &input_parameter_map)

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

- wps/tempus_services.cc

5.20 WPS::GetOptionsDescService Class Reference

Inheritance diagram for WPS::GetOptionsDescService:



Public Member Functions

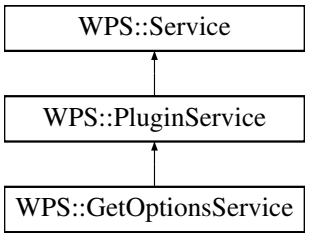
- Service::ParameterMap & **execute** (ParameterMap &input_parameter_map)

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

- wps/tempus_services.cc

5.21 WPS::GetOptionsService Class Reference

Inheritance diagram for WPS::GetOptionsService:



Public Member Functions

- Service::ParameterMap & **execute** (ParameterMap &input_parameter_map)

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

- wps/tempus_services.cc

5.22 Tempus::Multimodal::Graph Struct Reference

`#include <multimodal_graph.hh>`

Public Types

- typedef std::map< db_id_t, PublicTransport::Network > NetworkMap
- typedef std::map< db_id_t, PublicTransport::Graph > PublicTransportGraphList
- typedef std::map< db_id_t, POI > PoiList
- typedef std::map< std::string, Tempus::db_id_t > **NameToId**

Public Attributes

- Road::Graph road
- NetworkMap **network_map**
- PublicTransportGraphList **public_transports**
- PoiList **pois**
- RoadTypes road_types
- TransportTypes **transport_types**
- NameToId **road_type_from_name**
- NameToId **transport_type_from_name**

5.22.1 Detailed Description

A MultimodalGraph is basically a Road::Graph associated with a list of PublicTransport::Graph

5.22.2 Member Typedef Documentation

5.22.2.1 typedef std::map<db_id_t, PublicTransport::Network> Tempus::Multimodal::Graph::NetworkMap

Public transport networks

5.22.2.2 typedef std::map<db_id_t, POI> Tempus::Multimodal::Graph::PoiList

Point of interests

5.22.2.3 typedef std::map<db_id_t, PublicTransport::Graph> Tempus::Multimodal::Graph::PublicTransportGraphList

Public transports graphs network_id -> PublicTransport::Graph

5.22.3 Member Data Documentation

5.22.3.1 Road::Graph Tempus::Multimodal::Graph::road

The road graph

5.22.3.2 RoadTypes Tempus::Multimodal::Graph::road_types

Variables used to store constants.

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

- core/multimodal_graph.hh

5.23 boost::graph_traits< Tempus::Multimodal::Graph > Struct Template Reference

Public Types

- typedef Tempus::Multimodal::Vertex **vertex_descriptor**
- typedef Tempus::Multimodal::Edge **edge_descriptor**
- typedef Tempus::Multimodal::OutEdgeIterator **out_edge_iterator**

- typedef Tempus::Multimodal::VertexIterator **vertex_iterator**
- typedef Tempus::Multimodal::EdgeIterator **edge_iterator**
- typedef directed_tag **directed_category**
- typedef disallow_parallel_edge_tag **edge_parallel_category**
- typedef incidence_graph_tag **traversal_category**
- typedef size_t **vertices_size_type**
- typedef size_t **edges_size_type**
- typedef size_t **degree_size_type**

Static Public Member Functions

- static vertex_descriptor **null_vertex** ()

template<> struct boost::graph_traits< Tempus::Multimodal::Graph >

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

- core/multimodal_graph.hh

5.24 wps_client::HttpCgiConnection Class Reference

Public Member Functions

- def **__init__**
- def **reset**
- def **request**

Public Attributes

- **conn**
- **host**
- **url**

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

- wps/client/wps_client.py

5.25 Tempus::LengthCalculator Class Reference

Public Member Functions

- **LengthCalculator** (Db::Connection &db)
- double **operator()** (PublicTransport::Graph &graph, PublicTransport::Edge &e)

Protected Attributes

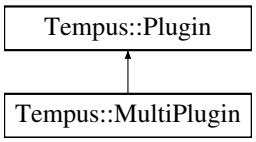
- Db::Connection & **db_**

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

- core/sample_pt_plugin.cc

5.26 Tempus::MultiPlugin Class Reference

Inheritance diagram for Tempus::MultiPlugin:



Public Member Functions

- **MultiPlugin** (Db::Connection &db)
- virtual void pre_process (Request &request) throw (std::invalid_argument)
- Multimodal::Vertex **vertex_from_road_node_id** (db_id_t id)
- virtual void process ()
- Result & result ()
- void cleanup ()

5.26.1 Member Function Documentation

5.26.1.1 void Tempus::MultiPlugin::cleanup ( ) [inline, virtual]

Cleanup method.

Reimplemented from Tempus::Plugin.

5.26.1.2 virtual void Tempus::MultiPlugin::pre_process (Request & *request*) throw (std::invalid_argument) [inline, virtual]

Pre-process the user request.

Parameters

in	*request*	The request to preprocess.

Exceptions

std::invalid_argument	Throws an instance of std::invalid_argument if the request cannot be processed by the current plugin.

Reimplemented from Tempus::Plugin.

5.26.1.3 virtual void Tempus::MultiPlugin::process ( ) [inline, virtual]

Process the last preprocessed user request. Must populates the 'result_' object.

Process the user request. Must populates the 'result_' object.

Reimplemented from Tempus::Plugin.

5.26.1.4 Result& Tempus::MultiPlugin::result ( ) [inline, virtual]

Result formatting

??? text formatting ?

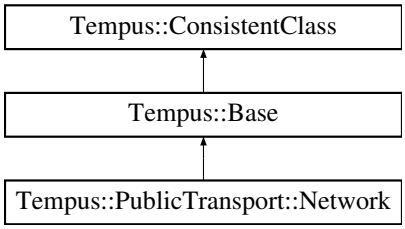
Reimplemented from Tempus::Plugin.

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

- core/sample_multi_plugin.cc

5.27 Tempus::PublicTransport::Network Struct Reference

Inheritance diagram for Tempus::PublicTransport::Network:



Public Attributes

- std::string **name**

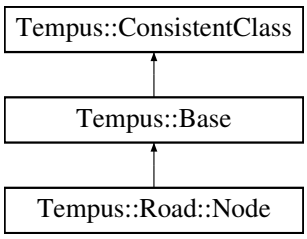
The documentation for this struct was generated from the following file:

- core/public_transport_graph.hh

5.28 Tempus::Road::Node Struct Reference

`#include <road_graph.hh>`

Inheritance diagram for Tempus::Road::Node:



Public Attributes

- Vertex vertex
- bool **is_junction**
- bool **is_bifurcation**

5.28.1 Detailed Description

Used as Vertex. Refers to the 'road_node' DB's table

5.28.2 Member Data Documentation

5.28.2.1 Vertex Tempus::Road::Node::vertex

This is a shortcut to the vertex index in the corresponding graph, if any. Needed to speedup access to a graph's vertex from a Node. Can be null

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

- core/road_graph.hh

5.29 Tempus::vertex_or_edge< G, Tag >::null_class Struct Reference

template<class G, class Tag> struct Tempus::vertex_or_edge< G, Tag >::null_class

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

- core/multimodal_graph.hh

5.30 Tempus::Plugin::OptionDescription Struct Reference

`#include <plugin.hh>`

Public Attributes

- OptionType **type**
- std::string **description**
- OptionValue **default_value**

5.30.1 Detailed Description

Plugin option description

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

- core/plugin.hh

5.31 Tempus::Plugin::OptionTypeFrom< T > Struct Template Reference

`#include <plugin.hh>`

Static Public Attributes

- static const OptionType **type**

5.31.1 Detailed Description

template<typename T>struct Tempus::Plugin::OptionTypeFrom< T >

Conversion from a C++ type to an OptionType. (Uses template specialization)

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

- core/plugin.hh

5.32 Tempus::Plugin::OptionTypeFrom< bool > Struct Template Reference

Static Public Attributes

- static const OptionType **type** = Plugin::BoolOption

template<> struct Tempus::Plugin::OptionTypeFrom< bool >

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

- core/plugin.hh

5.33 Tempus::Plugin::OptionTypeFrom< float > Struct Template Reference

Static Public Attributes

- static const OptionType **type** = Plugin::FloatOption

template<> struct Tempus::Plugin::OptionTypeFrom< float >

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

- core/plugin.hh

5.34 Tempus::Plugin::OptionTypeFrom< int > Struct Template Reference

Static Public Attributes

- static const OptionType **type** = Plugin::IntOption

template<> struct Tempus::Plugin::OptionTypeFrom< int >

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

- core/plugin.hh

5.35 Tempus::Plugin::OptionTypeFrom< std::string > Struct Template Reference

Static Public Attributes

- static const OptionType **type** = Plugin::StringOption

template<> struct Tempus::Plugin::OptionTypeFrom< std::string >

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

- core/plugin.hh

5.36 Tempus::Multimodal::OutEdgeIterator Struct Reference

Public Member Functions

- **OutEdgeIterator** (const Multimodal::Graph &graph, Multimodal::Vertex source)
- void **to_end** ()
- Multimodal::Edge & **dereference** () const
- void **increment** ()
- bool **equal** (const OutEdgeIterator &v) const

Protected Attributes

- Multimodal::Vertex **source_**
- const Multimodal::Graph * **graph_**
- Multimodal::Edge **edge_**
- Road::OutEdgeIterator **road_it_**
- Road::OutEdgeIterator **road_it_end_**
- PublicTransport::OutEdgeIterator **pt_it_**
- PublicTransport::OutEdgeIterator **pt_it_end_**
- size_t **stop2road_connection_**
- int **road2stop_connection_**
- int **road2poi_connection_**
- int **poi2road_connection_**

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

- core/multimodal_graph.hh
- core/multimodal_graph.cc

5.37 WPS::Service::ParameterSchema Struct Reference

Public Attributes

- std::string **schema**
- bool **is_complex**

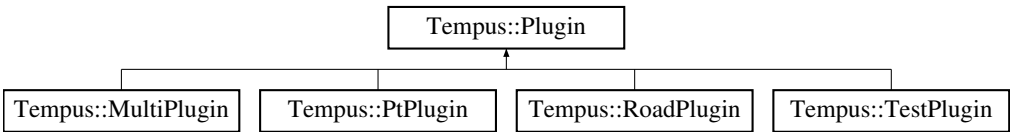
The documentation for this struct was generated from the following file:

- wps/wps_service.hh

5.38 Tempus::Plugin Class Reference

`#include <plugin.hh>`

Inheritance diagram for Tempus::Plugin:



Classes

- struct OptionDescription
- struct OptionTypeFrom
- struct OptionTypeFrom< bool >
- struct OptionTypeFrom< float >
- struct OptionTypeFrom< int >
- struct OptionTypeFrom< std::string >

Public Types

- enum OptionType { **BoolOption**, **IntOption**, **FloatOption**, **StringOption** }
- enum **AccessType** {

  InitAccess, **DiscoverAccess**, **ExamineAccess**, **EdgeRelaxedAccess**,

  EdgeNotRelaxedAccess, **EdgeMinimizedAccess**, **EdgeNotMinimizedAccess**, **TreeEdgeAccess**,

  NonTreeEdgeAccess, **BackEdgeAccess**, **ForwardOrCrossEdgeAccess**, **StartAccess**,

  FinishAccess, **GrayTargetAccess**, **BlackTargetAccess** }
- typedef std::map< std::string, Plugin * > PluginList
- typedef boost::any **OptionValue**
- typedef std::map< std::string, OptionValue > **OptionValueList**
- typedef std::map< std::string, OptionDescription > **OptionDescriptionList**
- typedef boost::any MetricValue
- typedef std::map< std::string, MetricValue > MetricValueList

Public Member Functions

- template<class T >
  void declare_option (const std::string &name, const std::string &description, T default_value)
- OptionDescriptionList & option_descriptions ()

- OptionValueList & **options** ()
- template<class T >
  void set_option (const std::string &name, const T &value)
- void set_option_from_string (const std::string &name, const std::string &value)
- std::string option_to_string (const std::string &name)
- template<class T >
  void get_option (const std::string &name, T &value)
- template<class T >
  T get_option (const std::string &name)
- MetricValueList & metrics ()
- std::string metric_to_string (const std::string &name)
- std::string name () const
- Plugin (const std::string &name, Db::Connection &db)
- virtual ~Plugin ()
- virtual void post_build ()
- virtual void validate ()
- virtual void road_vertex_accessor (Road::Vertex v, int access_type)
- virtual void **road_edge_accessor** (Road::Edge e, int access_type)
- virtual void **pt_vertex_accessor** (PublicTransport::Vertex v, int access_type)
- virtual void **pt_edge_accessor** (PublicTransport::Edge e, int access_type)
- virtual void **vertex_accessor** (Multimodal::Vertex v, int access_type)
- virtual void **edge_accessor** (Multimodal::Edge e, int access_type)
- virtual void cycle ()
- virtual void pre_process (Request &request) throw (std::invalid_argument)
- virtual void process ()
- virtual void post_process ()
- virtual Result & result ()
- virtual void cleanup ()

Static Public Member Functions

- static Plugin * load (const std::string &dll_name)
- static void unload (Plugin *plugin)
- static PluginList & **plugin_list** ()

Protected Attributes

- Multimodal::Graph & graph_
- Request request_
- Result result_
- std::string name_

  Name of this plugin.
- Db::Connection & db_

  Db connection.
- void * module_

The concrete plugin handler (HMODULE or void)*

- OptionDescriptionList options_descriptions_

  Plugin option management.
- OptionValueList **options_**
- MetricValueList **metrics_**

Static Protected Attributes

- static PluginList **plugin_list_**

5.38.1 Detailed Description

Base class that has to be derived in plugins

A Tempus plugin is made of :

- some user-defined options
- some callback functions called when user requests are processed
- some performance metrics

5.38.2 Member Typedef Documentation

5.38.2.1 typedef boost::any Tempus::Plugin::MetricValue

A metric is also a boost::any

5.38.2.2 typedef std::map<std::string, MetricValue> Tempus::Plugin::MetricValueList

Metric name -> value

5.38.2.3 typedef std::map<std::string, Plugin*> Tempus::Plugin::PluginList

Access to global plugin list

5.38.3 Member Enumeration Documentation

5.38.3.1 enum Tempus::Plugin::OptionType

Plugin option type

5.38.4 Constructor & Destructor Documentation

5.38.4.1 Tempus::Plugin::Plugin (const std::string & *name,* Db::Connection & *db*)

Called when the plugin is loaded into memory (install)

5.38.4.2 virtual Tempus::Plugin::~Plugin ( ) [inline, virtual]

Called when the plugin is unloaded from memory (uninstall)

5.38.5 Member Function Documentation

5.38.5.1 void Tempus::Plugin::cleanup ( ) [virtual]

Cleanup method.

Reimplemented in Tempus::MultiPlugin, Tempus::PtPlugin, and Tempus::RoadPlugin.

5.38.5.2 void Tempus::Plugin::cycle ( ) [virtual]

Cycle

5.38.5.3 template<class T > void Tempus::Plugin::declare_option (const std::string & *name,* const std::string & *description,* T *default_value*) [inline]

Method used by a plugin to declare an option

5.38.5.4 template<class T > void Tempus::Plugin::get_option (const std::string & *name,* T & *value*) [inline]

Method used to get an option value

5.38.5.5 template<class T > T Tempus::Plugin::get_option (const std::string & *name*) [inline]

Method used to get an option value, alternative signature.

5.38.5.6 Plugin * Tempus::Plugin::load (const std::string & *dll_name*) [static]

Static function used to load a plugin from disk

5.38.5.7 std::string Tempus::Plugin::metric_to_string (const std::string & *name*)

Converts a metric value to a string

5.38.5.8 MetricValueList& Tempus::Plugin::metrics ( ) [inline]

Access to metric list

5.38.5.9 std::string Tempus::Plugin::name ( ) const [inline]

Name accessor

5.38.5.10 OptionDescriptionList& Tempus::Plugin::option_descriptions ( ) [inline]

Option descriptions accessor

5.38.5.11 std::string Tempus::Plugin::option_to_string (const std::string & *name*)

Method used to get a string from an option value

5.38.5.12 void Tempus::Plugin::post_build ( ) [virtual]

Called after graphs have been built in memory.

Reimplemented in Tempus::RoadPlugin.

5.38.5.13 void Tempus::Plugin::post_process ( ) [virtual]

Post-process the user request.

5.38.5.14 void Tempus::Plugin::pre_process (Request & *request*) throw (std::invalid_argument) [virtual]

Pre-process the user request.

Parameters

in	*request*	The request to preprocess.

Exceptions

std::invalid_argument	Throws an instance of std::invalid_argument if the request cannot be processed by the current plugin.

Reimplemented in Tempus::MultiPlugin, Tempus::PtPlugin, Tempus::RoadPlugin, and Tempus::TestPlugin.

5.38.5.15 void Tempus::Plugin::process ( ) [virtual]

Process the last preprocessed user request. Must populates the 'result_' object.

Process the user request. Must populates the 'result_' object.

Reimplemented in Tempus::MultiPlugin, Tempus::PtPlugin, and Tempus::RoadPlugin.

5.38.5.16 Result & Tempus::Plugin::result ( ) [virtual]

Result formatting

??? text formatting ?

Reimplemented in Tempus::MultiPlugin, Tempus::PtPlugin, and Tempus::RoadPlugin.

5.38.5.17 virtual void Tempus::Plugin::road_vertex_accessor (Road::Vertex *v,* int *access_type*) [inline, virtual]

Acessors methods. They can be called on graph traversals. A Plugin is made compatible with a boost::visitor by means of a PluginGraphVisitor

Reimplemented in Tempus::RoadPlugin.

5.38.5.18 template<class T > void Tempus::Plugin::set_option (const std::string & *name,* const T & *value*) [inline]

Method used to set an option value

5.38.5.19 void Tempus::Plugin::set_option_from_string (const std::string & *name,* const std::string & *value*)

Method used to set an option value from a string. Conversions are made, based on the option description

5.38.5.20 void Tempus::Plugin::unload (Plugin * *plugin*) [static]

Static funtion used to unload a plugin

We cannot call delete directly on the plugin pointer, since it has been allocated from within another DLL.

5.38.5.21 void Tempus::Plugin::validate ( ) [virtual]

Called in order to validate the in-memory structure.

5.38.6 Member Data Documentation

5.38.6.1 Multimodal::Graph& Tempus::Plugin::graph_ [protected]

Graph extracted from the database

5.38.6.2 Request Tempus::Plugin::request_ [protected]

User request

5.38.6.3 Result Tempus::Plugin::result_ [protected]

Result

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

- core/plugin.hh
- core/plugin.cc

5.39 Tempus::PluginGraphVisitorHelper< Graph, VertexAccessorFunction, EdgeAccessorFunction > Class Template Reference

`#include <plugin.hh>`

Public Types

- typedef boost::graph_traits< Graph >::vertex_descriptor **VDescriptor**
- typedef boost::graph_traits< Graph >::edge_descriptor **EDescriptor**

Public Member Functions

- **PluginGraphVisitorHelper** (Plugin *plugin)
- void **initialize_vertex** (VDescriptor v, const Graph &graph)
- void **examine_vertex** (VDescriptor v, const Graph &graph)
- void **discover_vertex** (VDescriptor v, const Graph &graph)
- void **start_vertex** (VDescriptor v, const Graph &graph)
- void **finish_vertex** (VDescriptor v, const Graph &graph)
- void **examine_edge** (EDescriptor e, const Graph &graph)
- void **tree_edge** (EDescriptor e, const Graph &graph)
- void **non_tree_edge** (EDescriptor e, const Graph &graph)
- void **back_edge** (EDescriptor e, const Graph &graph)
- void **gray_target** (EDescriptor e, const Graph &graph)
- void **black_target** (EDescriptor e, const Graph &graph)

- void **forward_or_cross_edge** (EDescriptor e, const Graph &graph)
- void **edge_relaxed** (EDescriptor e, const Graph &graph)
- void **edge_not_relaxed** (EDescriptor e, const Graph &graph)
- void **edge_minimized** (EDescriptor e, const Graph &graph)
- void **edge_not_minimized** (EDescriptor e, const Graph &graph)

Protected Attributes

- Plugin * **plugin_**

5.39.1 Detailed Description

template<class Graph, void(Plugin::*)(typename boost::graph_traits< Graph >::vertex_descriptor, int) VertexAccessorFunction, void(Plugin::*)(typename boost::graph_traits< Graph >::edge_descriptor, int) EdgeAccessorFunction>class Tempus::PluginGraphVisitorHelper< Graph, VertexAccessorFunction, EdgeAccessorFunction >

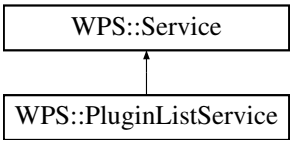
Class used as a boost::visitor. This is a proxy to Plugin::xxx_accessor methods. It may be used as implementation of any kind of boost::graph visitors (BFS, DFS, Dijkstra, A*, Bellman-Ford)

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

- core/plugin.hh

5.40 WPS::PluginListService Class Reference

Inheritance diagram for WPS::PluginListService:



Public Member Functions

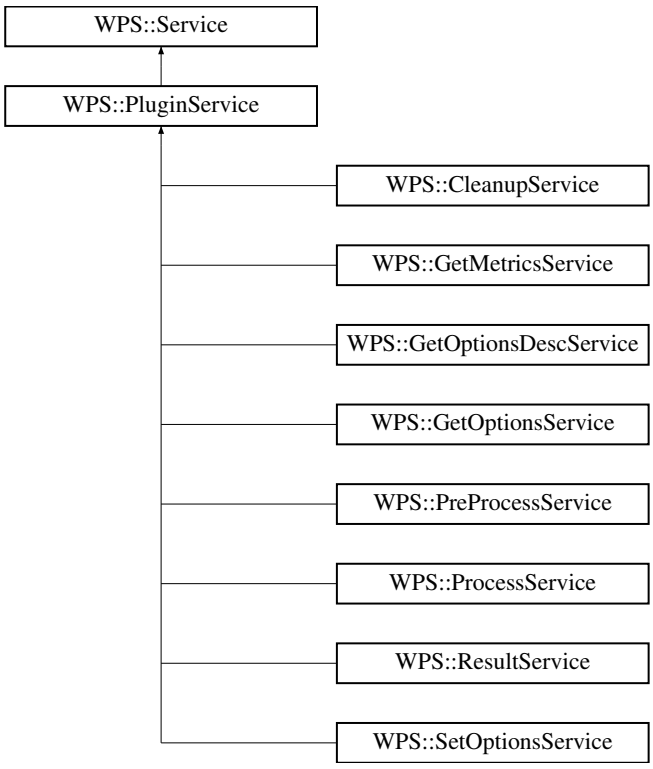
- Service::ParameterMap & **execute** (ParameterMap &input_parameter_map)

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

- wps/tempus_services.cc

5.41 WPS::PluginService Class Reference

Inheritance diagram for WPS::PluginService:



Public Member Functions

- **PluginService** (const std::string &name)
- Plugin * **get_plugin** (ParameterMap &input_parameters)

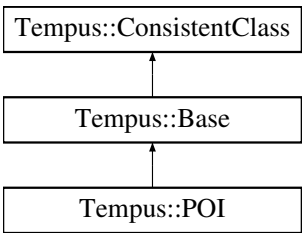
The documentation for this class was generated from the following file:

- wps/tempus_services.cc

5.42 Tempus::POI Struct Reference

`#include <road_graph.hh>`

Inheritance diagram for Tempus::POI:



Public Types

- enum **PoiType** {

  TypeCarPark = 1, **TypeSharedCarPoint**, **TypeCyclePark**, **TypeSharedCycle-Point**,

  TypeUserPOI }

Public Attributes

- int **poi_type**
- std::string **name**
- int parking_transport_type

  bitfield of TransportTypeId
- Road::Edge road_section
- double **abscissa_road_section**

5.42.1 Detailed Description

refers to the 'poi' DB's table

5.42.2 Member Data Documentation

5.42.2.1 Road::Edge Tempus::POI::road_section

Link to a road section. Must not be null.

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

- core/road_graph.hh

5.43 Tempus::Point2D Struct Reference

`#include <common.hh>`

Public Attributes

- double **x**
- double **y**

5.43.1 Detailed Description

2D Points

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

- core/common.hh

5.44 Tempus::PQImporter Class Reference

Public Member Functions

- **PQImporter** (const std::string &pg_options)
- Db::Result query (const std::string &query_str)
- void import_constants (Multimodal::Graph &graph, ProgressionCallback &callback=null_progression_callback)
- void import_graph (Multimodal::Graph &graph, ProgressionCallback &callback=null_progression_callback)
- Db::Connection & get_connection ()

Protected Attributes

- Db::Connection **connection_**

5.44.1 Member Function Documentation

5.44.1.1 Db::Connection& Tempus::PQImporter::get_connection ( ) [inline]

Access to underlying connection object

5.44.1.2 void Tempus::PQImporter::import_constants (Multimodal::Graph & *graph*, ProgressionCallback & *callback* = null_progression_callback)

Import constants (road, transports types) into global variables.

5.44.1.3 void Tempus::PQImporter::import_graph (Multimodal::Graph & *graph*, ProgressionCallback & *progression* = null_progression_callback)

Import the multimodal graph

Function used to import the road and public transport graphs from a PostgreSQL database.

5.44.1.4 Db::Result Tempus::PQImporter::query (const std::string & *query_str*)

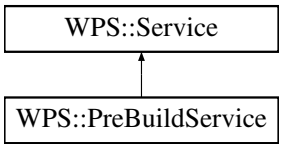
Query the database

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

- core/pgsql_importer.hh
- core/pgsql_importer.cc

5.45 WPS::PreBuildService Class Reference

Inheritance diagram for WPS::PreBuildService:



Public Member Functions

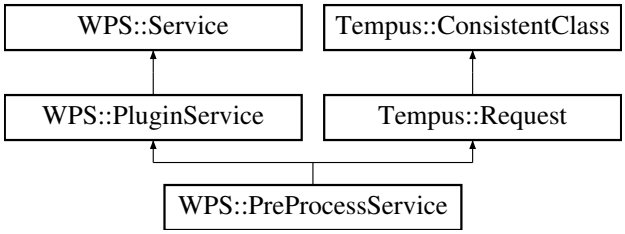
- Service::ParameterMap & **execute** (Service::ParameterMap &input_parameter_map)

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

- wps/tempus_services.cc

5.46 WPS::PreProcessService Class Reference

Inheritance diagram for WPS::PreProcessService:



Public Member Functions

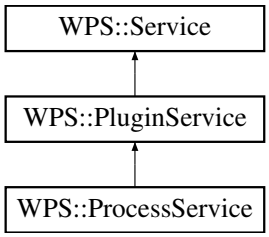
- void **parse_constraint** (xmlNode *node, Request::TimeConstraint &constraint)
- Service::ParameterMap & **execute** (ParameterMap &input_parameter_map)

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

- wps/tempus_services.cc

5.47 WPS::ProcessService Class Reference

Inheritance diagram for WPS::ProcessService:



Public Member Functions

- Service::ParameterMap & **execute** (ParameterMap &input_parameter_map)

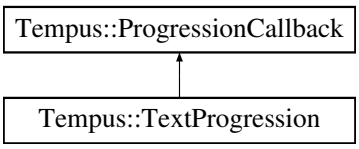
The documentation for this class was generated from the following file:

- wps/tempus_services.cc

5.48 Tempus::ProgressionCallback Class Reference

`#include <common.hh>`

Inheritance diagram for Tempus::ProgressionCallback:



Public Member Functions

- virtual void **operator()** (float, bool=false)

5.48.1 Detailed Description

Base class in charge of progression callback.

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

- core/common.hh

5.49 boost::property_traits< Tempus::FieldPropertyAccessor< Graph, Tag, T, Member > > Struct Template Reference

Public Types

- typedef T **value_type**
- typedef T & **reference**
- typedef Tempus::vertex_or_edge< Graph, Tag >::descriptor **key_type**
- typedef Tag **category**

template<class Graph, class Tag, class T, class Member> struct boost::property_traits< Tempus::FieldPropertyAccessor< Graph, Tag, T, Member > >

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

- core/multimodal_graph.hh

5.50 boost::property_traits< Tempus::FunctionPropertyAccessor< Graph, Tag, T, Function > > Struct Template Reference

Public Types

- typedef T **value_type**
- typedef T & **reference**

- typedef Tempus::vertex_or_edge< Graph, Tag >::descriptor **key_type**
- typedef Tag **category**

template<class Graph, class Tag, class T, class Function> struct boost::property_traits< Tempus::FunctionPropertyAccessor< Graph, Tag, T, Function > >

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

- core/multimodal_graph.hh

5.51 boost::property_traits< Tempus::Multimodal::EdgeIndexProperty > Struct Template Reference

Public Types

- typedef size_t **value_type**
- typedef size_t & **reference**
- typedef Tempus::Multimodal::Edge **key_type**
- typedef boost::edge_property_tag **category**

template<> struct boost::property_traits< Tempus::Multimodal::EdgeIndexProperty >

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

- core/multimodal_graph.hh

5.52 boost::property_traits< Tempus::Multimodal::VertexIndexProperty > Struct Template Reference

Public Types

- typedef size_t **value_type**
- typedef size_t & **reference**
- typedef Tempus::Multimodal::Vertex **key_type**
- typedef boost::vertex_property_tag **category**

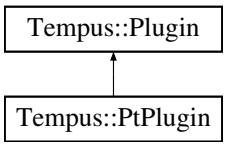
template<> struct boost::property_traits< Tempus::Multimodal::VertexIndexProperty >

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

- core/multimodal_graph.hh

5.53 Tempus::PtPlugin Class Reference

Inheritance diagram for Tempus::PtPlugin:



Public Member Functions

- **PtPlugin** (Db::Connection &db)
- virtual void pre_process (Request &request) throw (std::invalid_argument)
- virtual void **pt_edge_accessor** (PublicTransport::Edge e, int access_type)
- virtual void **pt_vertex_accessor** (PublicTransport::Vertex v, int access_type)
- virtual void process ()
- Result & result ()
- void cleanup ()

5.53.1 Member Function Documentation

5.53.1.1 void Tempus::PtPlugin::cleanup ( ) [inline, virtual]

Cleanup method.

Reimplemented from Tempus::Plugin.

5.53.1.2 virtual void Tempus::PtPlugin::pre_process (Request & *request*) throw (std::invalid_argument) [inline, virtual]

Pre-process the user request.

Parameters

in	*request*	The request to preprocess.

Exceptions

std::invalid_argument	Throws an instance of std::invalid_argument if the request cannot be processed by the current plugin.

Reimplemented from Tempus::Plugin.

5.53.1.3 virtual void Tempus::PtPlugin::process ( ) [inline, virtual]

Process the last preprocessed user request. Must populates the 'result_' object.

Process the user request. Must populates the 'result_' object.

Reimplemented from Tempus::Plugin.

5.53.1.4 Result& Tempus::PtPlugin::result ( ) [inline, virtual]

Result formatting

??? text formatting ?

Reimplemented from Tempus::Plugin.

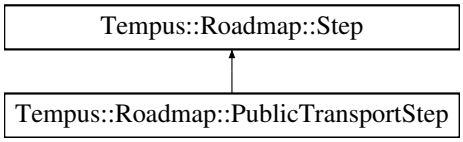
The documentation for this class was generated from the following file:

- core/sample_pt_plugin.cc

5.54 Tempus::Roadmap::PublicTransportStep Struct Reference

`#include <roadmap.hh>`

Inheritance diagram for Tempus::Roadmap::PublicTransportStep:



Public Attributes

- db_id_t **network_id**
- PublicTransport::Vertex **departure_stop**
- PublicTransport::Vertex **arrival_stop**
- db_id_t trip_id

  used to indicate the direction

5.54.1 Detailed Description

A Step made with a public transport

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

- core/roadmap.hh

5.55 WPS::Request Class Reference

`#include <wps_request.hh>`

Public Member Functions

- **Request** (std::streambuf *ins, std::streambuf *outs)
- int **process** ()
- int **print_error_status** (int status, const std::string &msg)
- int **print_exception** (const std::string &type, const std::string &msg)

Protected Attributes

- std::istream **ins_**
- std::ostream **outs_**

5.55.1 Detailed Description

WPS::Request

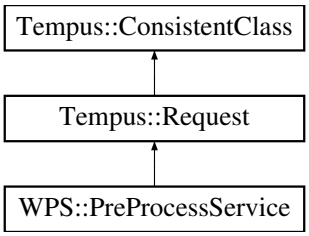
The documentation for this class was generated from the following files:

- wps/wps_request.hh
- wps/wps_request.cc

5.56 Tempus::Request Class Reference

`#include <request.hh>`

Inheritance diagram for Tempus::Request:



Classes

- struct Step
- struct TimeConstraint

Public Types

- typedef std::vector< Step > **StepList**

Public Member Functions

- Road::Vertex destination ()

Public Attributes

- StepList steps
- unsigned allowed_transport_types
- Road::Vertex parking_location
- std::vector< db_id_t > allowed_networks
- TimeConstraint departure_constraint
- Road::Vertex origin
- std::vector< int > optimizing_criteria

Protected Member Functions

- bool check_consistency_ ()

5.56.1 Detailed Description

A Request is used to model user requests to the planning engine.

5.56.2 Member Function Documentation

5.56.2.1 bool Tempus::Request::check_consistency_ ( ) [inline, protected, virtual]

Private method to override in derived classes. Does nothing by default.

Reimplemented from Tempus::ConsistentClass.

5.56.2.2 Road::Vertex Tempus::Request::destination ( ) [inline]

Shortcut to get the final destination (the last step)

5.56.3 Member Data Documentation

5.56.3.1 std::vector<db_id_t> Tempus::Request::allowed_networks

Public transport options: list of allowed networks

5.56.3.2 unsigned Tempus::Request::allowed_transport_types

Allowed transport types. It can be stored in an integer, since transport_type ID are powers of two.

5.56.3.3 TimeConstraint Tempus::Request::departure_constraint

Timeing constraint on the departure

5.56.3.4 std::vector<int> Tempus::Request::optimizing_criteria

Criteria to optimize. The list is ordered by criterion priority. Refers to a CostId (see common.hh)

5.56.3.5 Road::Vertex Tempus::Request::origin

Vertex origin of the request

5.56.3.6 Road::Vertex Tempus::Request::parking_location

Private vehicule option: parking location

5.56.3.7 StepList Tempus::Request::steps

Steps involved in the request. It has to be made at a minimum of an origin and a destination. It may include intermediary points.

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

- core/request.hh

5.57 Db::Result Class Reference

`#include <db.hh>`

Public Member Functions

- **Result** (PGresult *res)
- Result (const Result &r)
- Result & operator= (const Result &r)
- size_t size ()
- size_t columns ()
- RowValue operator[] (size_t idx)

Protected Member Functions

- void **dec_refs** () const
- void **inc_refs** () const

Protected Attributes

- PGresult * **res_**
- int **nrefs_**

5.57.1 Detailed Description

Class representing result of a query

5.57.2 Constructor & Destructor Documentation

5.57.2.1 Db::Result::Result (const Result & *r*) [inline]

Copy constructor

5.57.3 Member Function Documentation

5.57.3.1 size_t Db::Result::columns ( ) [inline]

Number of columns

5.57.3.2 Result& Db::Result::operator= (const Result & *r*) [inline]

Assignment operator. Deals with reference counting

5.57.3.3 RowValue Db::Result::operator[] (size_t *idx*) [inline]

Access to a row of a result, by row number

5.57.3.4 size_t Db::Result::size ( ) [inline]

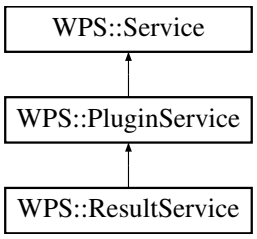
Number of rows

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

- core/db.hh

5.58 WPS::ResultService Class Reference

Inheritance diagram for WPS::ResultService:



Public Member Functions

- Service::ParameterMap & **execute** (ParameterMap &input_parameter_map)

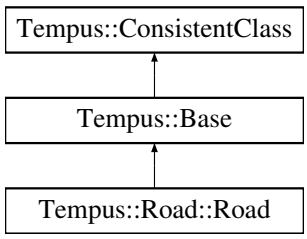
The documentation for this class was generated from the following file:

- wps/tempus_services.cc

5.59 Tempus::Road::Road Struct Reference

`#include <road_graph.hh>`

Inheritance diagram for Tempus::Road::Road:



Public Attributes

- std::vector< Edge > road_section
- double cost

5.59.1 Detailed Description

refers to the 'road_road' DB's table

5.59.2 Member Data Documentation

5.59.2.1 double Tempus::Road::Road::cost

-1 means infinite cost

5.59.2.2 std::vector<Edge> Tempus::Road::Road::road_section

Array of road sections

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

- core/road_graph.hh

5.60 Tempus::Roadmap Class Reference

`#include <roadmap.hh>`

Classes

- struct PublicTransportStep
- struct RoadStep
- struct Step

Public Types

- typedef std::vector< Step * > StepList
- typedef std::vector< Point2D > PointList

Public Attributes

- StepList **steps**
- Costs **total_costs**
- PointList **overview_path**

5.60.1 Detailed Description

A Roadmap is an object used to model steps involved in a multimodal route. It is a base for result values of a request.

5.60.2 Member Typedef Documentation

5.60.2.1 typedef std::vector<Point2D> Tempus::Roadmap::PointList

Optional overview path, which is designed for display purposes, and may be simplified

5.60.2.2 typedef std::vector<Step*> Tempus::Roadmap::StepList

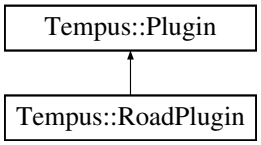
A Roadmap is a list of Step augmented with some total costs. Ownership : pointers are allocated by the caller but freed on Roadmap destruction

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

- core/roadmap.hh

5.61 Tempus::RoadPlugin Class Reference

Inheritance diagram for Tempus::RoadPlugin:



Public Member Functions

- **RoadPlugin** (Db::Connection &db)
- virtual void post_build ()
- virtual void pre_process (Request &request) throw (std::invalid_argument)
- virtual void road_vertex_accessor (Road::Vertex v, int access_type)
- virtual void process ()
- Result & result ()
- void cleanup ()

Protected Attributes

- bool **trace_vertex_**

5.61.1 Member Function Documentation

5.61.1.1 void Tempus::RoadPlugin::cleanup ( ) [inline, virtual]

Cleanup method.

Reimplemented from Tempus::Plugin.

5.61.1.2 virtual void Tempus::RoadPlugin::post_build ( ) [inline, virtual]

Called after graphs have been built in memory.

Reimplemented from Tempus::Plugin.

5.61.1.3 virtual void Tempus::RoadPlugin::pre_process (Request & *request*) throw (std::invalid_argument) [inline, virtual]

Pre-process the user request.

Parameters

in	*request*	The request to preprocess.

Exceptions

std::invalid_argument	Throws an instance of std::invalid_argument if the request cannot be processed by the current plugin.

Reimplemented from Tempus::Plugin.

5.61.1.4 virtual void Tempus::RoadPlugin::process ( ) [inline, virtual]

Process the last preprocessed user request. Must populates the 'result_' object.

Process the user request. Must populates the 'result_' object.

We define a property map that reads the 'length' (of type double) member of a Road::Section, which is the edge property of a Road::Graph

Visitor to be built on 'this'. This way, xxx_accessor methods will be called

Reimplemented from Tempus::Plugin.

5.61.1.5 Result& Tempus::RoadPlugin::result ( ) [inline, virtual]

Result formatting

??? text formatting ?

Reimplemented from Tempus::Plugin.

5.61.1.6 virtual void Tempus::RoadPlugin::road_vertex_accessor (Road::Vertex *v*, int *access_type*) [inline, virtual]

Acessors methods. They can be called on graph traversals. A Plugin is made compatible with a boost::visitor by means of a PluginGraphVisitor

Reimplemented from Tempus::Plugin.

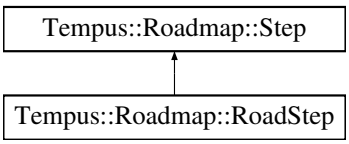
The documentation for this class was generated from the following file:

- core/sample_road_plugin.cc

5.62 Tempus::Roadmap::RoadStep Struct Reference

`#include <roadmap.hh>`

Inheritance diagram for Tempus::Roadmap::RoadStep:



Public Types

- enum EndMovement {
  GoAhead, **TurnLeft**, **TurnRight**, **UTurn**,
  RoundAboutEnter, FirstExit, **SecondExit**, **ThirdExit**,
  FourthExit, **FifthExit**, **SixthExit**, **YouAreArrived** = 999 }
  The movement that is to be done at the end of the section.

Public Attributes

- Road::Edge road_section
- Road::Edge road_direction
- double distance_km
- EndMovement **end_movement**

5.62.1 Detailed Description

A Step that occurs on the road, either by a pedestrian or a private vehicle

5.62.2 Member Enumeration Documentation

5.62.2.1 enum Tempus::Roadmap::RoadStep::EndMovement

The movement that is to be done at the end of the section.

Enumerator:

FirstExit in a roundabout

5.62.3 Member Data Documentation

5.62.3.1 double Tempus::Roadmap::RoadStep::distance_km

Distance to walk/drive (in km). -1 if we have to go until the end of the section

5.62.3.2 Road::Edge Tempus::Roadmap::RoadStep::road_direction

The road section where to go in the direction of

5.62.3.3 Road::Edge Tempus::Roadmap::RoadStep::road_section

The road section where to start from

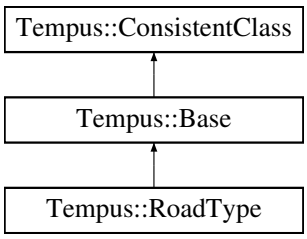
The documentation for this struct was generated from the following file:

- core/roadmap.hh

5.63 Tempus::RoadType Struct Reference

`#include <common.hh>`

Inheritance diagram for Tempus::RoadType:



Public Attributes

- std::string **name**

5.63.1 Detailed Description

Refers to tempus.road_type table

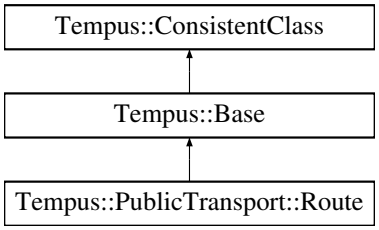
The documentation for this struct was generated from the following file:

- core/common.hh

5.64 Tempus::PublicTransport::Route Struct Reference

`#include <public_transport_graph.hh>`

Inheritance diagram for Tempus::PublicTransport::Route:



Public Types

- enum **RouteType** {
  TypeTram = 0, **TypeSubway**, **TypeRail**, **TypeBus**,
  TypeFerry, **TypeCableCar**, **TypeSuspendedCar**, **TypeFunicular** }

Public Attributes

- db_id_t network_id
- std::string **short_name**
- std::string **long_name**
- int **route_type**
- std::vector< Trip > **trips**

Protected Member Functions

- bool check_consistency_ ()

5.64.1 Detailed Description

refers to the 'pt_route' DB's table

5.64.2 Member Function Documentation

5.64.2.1 bool Tempus::PublicTransport::Route::check_consistency_ ( ) [inline, protected, virtual]

Private method to override in derived classes. Does nothing by default.

Reimplemented from Tempus::ConsistentClass.

5.64.3 Member Data Documentation

5.64.3.1 db_id_t Tempus::PublicTransport::Route::network_id

public transport network

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

- core/public_transport_graph.hh

5.65 Db::RowValue Class Reference

`#include <db.hh>`

Public Member Functions

- **RowValue** (PGresult *res, size_t nrow)
- Value operator[] (size_t fn)

Protected Attributes

- PGresult * **res_**
- size_t **nrow_**

5.65.1 Detailed Description

Class used to represent a row in a result.

5.65.2 Member Function Documentation

5.65.2.1 Value Db::RowValue::operator[] (size_t *fn*) [inline]

Access to a value by column number

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

- core/db.hh

5.66 scoped_ptr< T, deletion_fct > Class Template Reference

`#include <xml_helper.hh>`

Public Member Functions

- **scoped_ptr** (T *ptr)
- **scoped_ptr** (const scoped_ptr< T, deletion_fct > &p)
- scoped_ptr< T, deletion_fct > & **operator=** (const scoped_ptr< T, deletion_fct > &p)
- T * **get** ()
- void **set** (T *ptr)

Protected Member Functions

- void **deleteme** ()

Protected Attributes

- T * **ptr_**

5.66.1 Detailed Description

template<class T, void deletion_fct>class scoped_ptr< T, deletion_fct >

Helper functions around libxml Helper class designed to hold already-allocated pointers and call a deletion function when the object is out of scope. This is the way libxml works: it returns allocated pointers that have to be freed by the caller

There is no reference counting. Objets are "moved" from instances (as boost::auto_ptr does) For example a = b transfers ownership from b to a and b is set to null

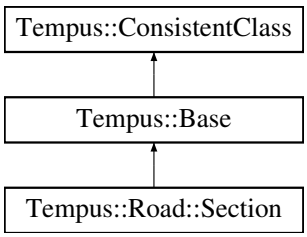
The documentation for this class was generated from the following file:

- wps/xml_helper.hh

5.67 Tempus::Road::Section Struct Reference

`#include <road_graph.hh>`

Inheritance diagram for Tempus::Road::Section:



Public Attributes

- Edge edge
- db_id_t **road_type**
- int transport_type_ft

  bitfield of TransportTypeId
- int transport_type_tf

  bitfield of TransportTypeId
- double **length**
- double **car_speed_limit**
- double **car_average_speed**
- double **bus_average_speed**
- std::string **road_name**
- std::string **address_left_side**
- std::string **address_right_side**
- int **lane**
- bool **is_roundabout**
- bool **is_bridge**
- bool **is_tunnel**
- bool **is_ramp**
- bool **is_tollway**
- std::vector< PublicTransport::Stop * > stops
- std::vector< POI * > pois

5.67.1 Detailed Description

Used as Directed Edge. Refers to the 'road_section' DB's table

5.67.2 Member Data Documentation

5.67.2.1 Edge Tempus::Road::Section::edge

This is a shortcut to the edge index in the corresponding graph, if any. Needed to speedup access to a graph's edge from a Section. Can be null

5.67.2.2 std::vector<POI*> Tempus::Road::Section::pois

List of Point Of Interests attached to this road section

5.67.2.3 std::vector< PublicTransport::Stop* > Tempus::Road::Section::stops

List of public transport stops, attached to this road section

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

- core/road_graph.hh

5.68 Tempus::PublicTransport::Section Struct Reference

`#include <public_transport_graph.hh>`

Public Attributes

- Edge edge
- const Graph * **graph**
- db_id_t **stop_from**
- db_id_t **stop_to**
- db_id_t network_id

  must not be null

5.68.1 Detailed Description

used as an Edge in a PublicTransportGraph

5.68.2 Member Data Documentation

5.68.2.1 Edge Tempus::PublicTransport::Section::edge

This is a shortcut to the edge index in the corresponding graph, if any. Needed to speedup access to a graph's edge from a Section Can be null

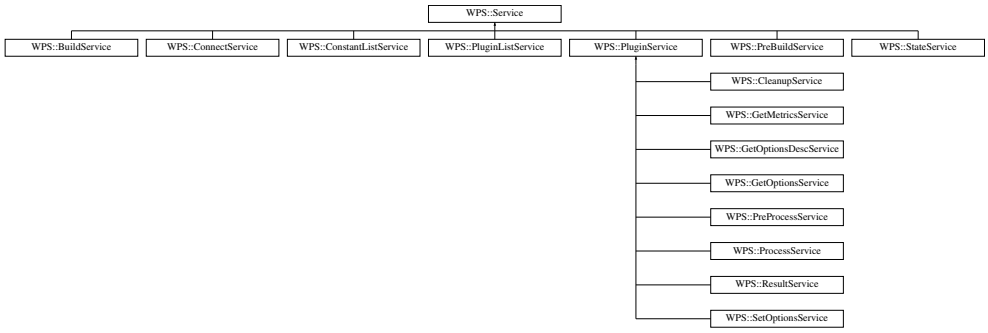
The documentation for this struct was generated from the following file:

- core/public_transport_graph.hh

5.69 WPS::Service Class Reference

`#include <wps_service.hh>`

Inheritance diagram for WPS::Service:



Classes

- struct ParameterSchema

Public Types

- typedef std::map< std::string, xmlNode * > **ParameterMap**

Public Member Functions

- **Service** (const std::string &name)
- void parse_xml_parameters (ParameterMap &input_parameter_map)
- virtual ParameterMap & **execute** (ParameterMap &input_parameter_map)
- std::ostream & get_xml_description (std::ostream &out)
- std::ostream & get_xml_execute_response (std::ostream &out)

Static Public Member Functions

- static Service * get_service (const std::string &name)
- static bool exists (const std::string &name)
- static std::ostream & get_xml_capabilities (std::ostream &out)

Protected Types

- typedef std::map< std::string, ParameterSchema > **SchemaMap**

Protected Member Functions

- virtual void check_parameters (ParameterMap &parameter_map, SchemaMap &schema_map)
- void add_input_parameter (const std::string &name, const std::string &schema, bool is_complex=true)
- void add_output_parameter (const std::string &name, const std::string &schema, bool is_complex=true)

Protected Attributes

- SchemaMap **input_parameter_schema_**
- SchemaMap **output_parameter_schema_**
- std::string **name_**
- ParameterMap output_parameters_

Static Protected Attributes

- static std::map< std::string, Service * > * services_ = 0

5.69.1 Detailed Description

Function callable from a WPS 'Execute' operation

5.69.2 Member Function Documentation

5.69.2.1 void WPS::Service::add_input_parameter (const std::string & *name,* const std::string & *schema,* bool *is_complex* = `true`) [inline, protected]

Adds an input parameter definition. To be called by derived classes in their constructor

5.69.2.2 void WPS::Service::add_output_parameter (const std::string & *name,* const std::string & *schema,* bool *is_complex* = `true`) [inline, protected]

Adds an output parameter definition. To be called by derived classes in their constructor

5.69.2.3 void WPS::Service::check_parameters (ParameterMap & *parameter_map,* SchemaMap & *schema_map*) [protected, virtual]

Check parameters against their XML schemas

5.69.2.4 static bool WPS::Service::exists (const std::string & *name*) [inline, static]

Global service map interface: tests if a service exists

5.69.2.5 Service * WPS::Service::get_service (const std::string & *name*) [static]

Global service map interface: returns a Service* based on a service name

5.69.2.6 std::ostream & WPS::Service::get_xml_capabilities (std::ostream & *out*) [static]

Global service map interface: returns an XML string that conforms to a 'GetCapabilities' operation

5.69.2.7 std::ostream & WPS::Service::get_xml_description (std::ostream & *out*)

Returns an XML string that conforms to a DescribeProcess operation

5.69.2.8 std::ostream & WPS::Service::get_xml_execute_response (std::ostream & *out*)

Returns an XML string that represents results of an Execute operation

5.69.2.9 void WPS::Service::parse_xml_parameters (ParameterMap & *input_parameter_map*)

Extract input parameters

5.69.3 Member Data Documentation

5.69.3.1 ParameterMap WPS::Service::output_parameters_ [protected]

Output parameters

5.69.3.2 std::map< std::string, Service * > * WPS::Service::services_ = 0 [static, protected]

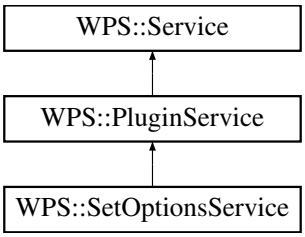
A global map of services

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

- wps/wps_service.hh
- wps/wps_service.cc

5.70 WPS::SetOptionsService Class Reference

Inheritance diagram for WPS::SetOptionsService:



Public Member Functions

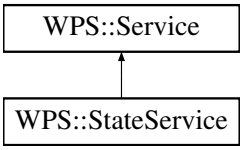
- Service::ParameterMap & **execute** (ParameterMap &input_parameter_map)

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

- wps/tempus_services.cc

5.71 WPS::StateService Class Reference

Inheritance diagram for WPS::StateService:



Public Member Functions

- Service::ParameterMap & **execute** (Service::ParameterMap &input_parameter_map)

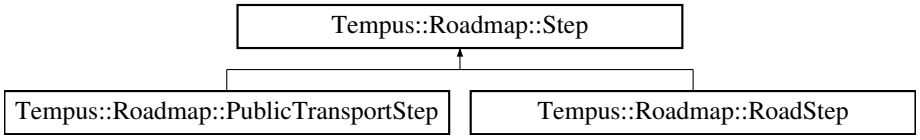
The documentation for this class was generated from the following file:

- wps/tempus_services.cc

5.72 Tempus::Roadmap::Step Struct Reference

`#include <roadmap.hh>`

Inheritance diagram for Tempus::Roadmap::Step:



Public Types

- enum **StepType** { **VertexStep** = 0, **RoadStep**, **PublicTransportStep** }

Public Member Functions

- **Step** (StepType type)

Public Attributes

- StepType **step_type**
- Costs **costs**

5.72.1 Detailed Description

A Step is a part of a route, where the transport type is constant This a generic class

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

- core/roadmap.hh

5.73 Tempus::Request::Step Struct Reference

`#include <request.hh>`

Public Attributes

- Road::Vertex **destination**
- TimeConstraint **constraint**
- bool private_vehicule_at_destination

5.73.1 Detailed Description

Class used to represent destinations of a request and constraints of the step

5.73.2 Member Data Documentation

5.73.2.1 bool Tempus::Request::Step::private_vehicule_at_destination

Whether the private vehicule must reach the destination

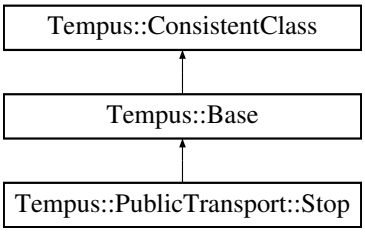
The documentation for this struct was generated from the following file:

- core/request.hh

5.74 Tempus::PublicTransport::Stop Struct Reference

`#include <public_transport_graph.hh>`

Inheritance diagram for Tempus::PublicTransport::Stop:



Public Attributes

- Vertex vertex
- const Graph * **graph**
- std::string **name**
- bool **is_station**
- Vertex parent_station
- bool **has_parent**
- Road::Edge road_section
- double **abscissa_road_section**
- int zone_id

5.74.1 Detailed Description

Used as a vertex in a PublicTransportGraph. Refers to the 'pt_stop' DB's table

5.74.2 Member Data Documentation

5.74.2.1 Vertex Tempus::PublicTransport::Stop::parent_station

link to a possible parent station, or null

5.74.2.2 Road::Edge Tempus::PublicTransport::Stop::road_section

link to a road section must not be null

5.74.2.3 Vertex Tempus::PublicTransport::Stop::vertex

This is a shortcut to the vertex index in the corresponding graph, if any. Needed to speedup access to a graph's vertex from a Node. Can be null

5.74.2.4 int Tempus::PublicTransport::Stop::zone_id

Fare zone ID of this stop

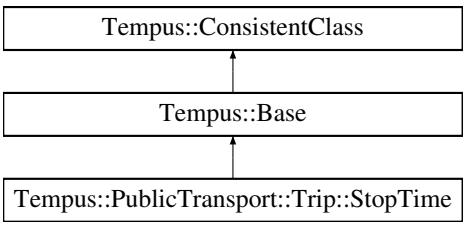
The documentation for this struct was generated from the following file:

- core/public_transport_graph.hh

5.75 Tempus::PublicTransport::Trip::StopTime Struct Reference

`#include <public_transport_graph.hh>`

Inheritance diagram for Tempus::PublicTransport::Trip::StopTime:



Public Attributes

- PublicTransport::Vertex stop
- Time **arrival_time**
- Time **departure_time**
- std::string **stop_headsign**
- int **pickup_type**
- int **drop_off_type**
- double **shape_dist_traveled**

5.75.1 Detailed Description

Refers to the 'pt_stop_time' table

5.75.2 Member Data Documentation

5.75.2.1 PublicTransport::Vertex Tempus::PublicTransport::Trip::StopTime::stop

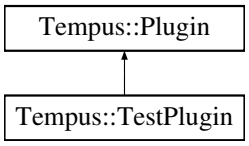
Link to the Stop. Must not be null. Represents the link part of the "stop_sequence" field

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

- core/public_transport_graph.hh

5.76 Tempus::TestPlugin Class Reference

Inheritance diagram for Tempus::TestPlugin:



Public Member Functions

- **TestPlugin** (Db::Connection &db)
- virtual void pre_process (Request &request) throw (std::invalid_argument)

5.76.1 Member Function Documentation

5.76.1.1 virtual void Tempus::TestPlugin::pre_process (Request & *request*) throw (std::invalid_argument) [inline, virtual]

Pre-process the user request.

Parameters

in	*request*	The request to preprocess.

Exceptions

std::invalid_argument	Throws an instance of std::invalid_argument if the request cannot be processed by the current plugin.

Reimplemented from Tempus::Plugin.

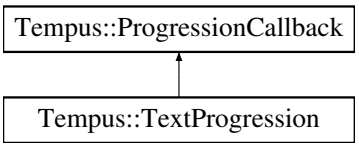
The documentation for this class was generated from the following file:

- core/test_plugin.cc

5.77 Tempus::TextProgression Struct Reference

`#include <common.hh>`

Inheritance diagram for Tempus::TextProgression:



Public Member Functions

- **TextProgression** (int N=50)
- virtual void **operator()** (float percent, bool finished)

Protected Attributes

- int **N_**
- int **old_N_**

5.77.1 Detailed Description

Simple progession processing: text based progression bar.

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

- core/common.hh
- core/common.cc

5.78 Tempus::Time Struct Reference

`#include <common.hh>`

Public Attributes

- long **n_secs**

5.78.1 Detailed Description

Time is the number of seconds since 00:00.

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

- core/common.hh

5.79 Tempus::Request::TimeConstraint Struct Reference

Public Types

- enum **TimeConstraintType** { **NoConstraint** = 0, **ConstraintBefore**, **ConstraintAfter** }

Public Attributes

- int type

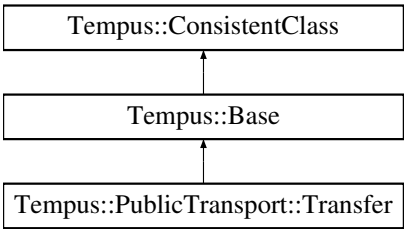
  TimeConstraintType.
- DateTime **date_time**

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

- core/request.hh

5.80 Tempus::PublicTransport::Transfer Struct Reference

Inheritance diagram for Tempus::PublicTransport::Transfer:



Public Types

- enum **TranferType** { **NormalTransfer** = 0, **TimedTransfer**, **MinimalTimedTransfer**, **ImpossibleTransfer** }

Public Attributes

- Vertex from_stop
- Vertex **to_stop**
- int **transfer_type**
- int min_transfer_time

Protected Member Functions

- bool check_consistency_ ()

5.80.1 Member Function Documentation

5.80.1.1 bool Tempus::PublicTransport::Transfer::check_consistency_ ( ) [inline, protected, virtual]

Private method to override in derived classes. Does nothing by default.

Reimplemented from Tempus::ConsistentClass.

5.80.2 Member Data Documentation

5.80.2.1 Vertex Tempus::PublicTransport::Transfer::from_stop

Link between two stops. Must not be null

5.80.2.2 int Tempus::PublicTransport::Transfer::min_transfer_time

Must be positive not null. Expressed in seconds

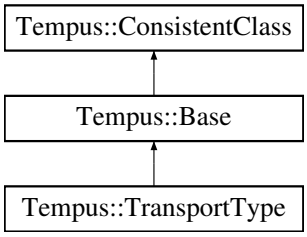
The documentation for this struct was generated from the following file:

- core/public_transport_graph.hh

5.81 Tempus::TransportType Struct Reference

`#include <common.hh>`

Inheritance diagram for Tempus::TransportType:



Public Attributes

- db_id_t **parent_id**
- std::string **name**

- bool **need_parking**
- bool **need_station**
- bool **need_return**

Protected Member Functions

- bool check_consistency_ ()

5.81.1 Detailed Description

Refers to tempus.transport_type table

5.81.2 Member Function Documentation

5.81.2.1 bool Tempus::TransportType::check_consistency_ ( ) [inline, protected, virtual]

Private method to override in derived classes. Does nothing by default.

x is a power of two if (x & (x - 1)) is 0

Reimplemented from Tempus::ConsistentClass.

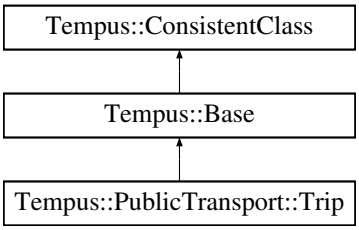
The documentation for this struct was generated from the following file:

- core/common.hh

5.82 Tempus::PublicTransport::Trip Struct Reference

`#include <public_transport_graph.hh>`

Inheritance diagram for Tempus::PublicTransport::Trip:



Classes

- struct Frequency
- struct StopTime

Public Types

- typedef std::list< std::vector< StopTime > > StopTimes
- typedef std::list< Frequency > **Frequencies**

Public Attributes

- std::string **short_name**
- StopTimes stop_times
- Frequencies frequencies
- Calendar * service

Protected Member Functions

- bool check_consistency_ ()

5.82.1 Detailed Description

Trip, Route, StopTime and Frequencies classes

5.82.2 Member Typedef Documentation

5.82.2.1 typedef std::list< std::vector< StopTime > > Tempus::PublicTransport::Trip::StopTimes

This is the definition of a list of stop times for a trip. The list of stop times has to be ordered to represent the sequence of stops (based on the "stop_sequence" field of the corresponding "stop_times" table

5.82.3 Member Function Documentation

5.82.3.1 bool Tempus::PublicTransport::Trip::check_consistency_ ( ) [inline, protected, virtual]

Private method to override in derived classes. Does nothing by default.

Reimplemented from Tempus::ConsistentClass.

5.82.4 Member Data Documentation

5.82.4.1 Frequencies Tempus::PublicTransport::Trip::frequencies

List of frequencies for this trip

5.82.4.2 Calendar* Tempus::PublicTransport::Trip::service

Link to the dates when service is available. Must not be null.

5.82.4.3 StopTimes Tempus::PublicTransport::Trip::stop_times

List of all stop times. Can be a subset of those stored in the database.

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

- core/public_transport_graph.hh

5.83 Db::Value Class Reference

`#include <db.hh>`

Public Member Functions

- **Value** (const char *value, size_t len, bool isnull)
- template<class T >
  T as ()
- template<class T >
  void operator>> (T &obj)
- bool is_null ()

Protected Attributes

- const char * **value_**
- size_t **len_**
- bool **isnull_**

5.83.1 Detailed Description

Class representing an atomic value stored in a database.

5.83.2 Member Function Documentation

5.83.2.1 double Db::Value::as< double > ( ) [inline]

This is the generic conversion operator. It calls stringstream conversion operators (slow!). Specialization can be introduced, or via a specialization of the stringstream::operator>>()

5.83.2.2 bool Db::Value::is_null ( ) [inline]

Tests if the underlying object is null

5.83.2.3 template<class T > void Db::Value::operator>> (T & *obj*) [inline]

Conversion operator. Does nothing if the underlying object is null (which is a special value in a database)

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

- core/db.hh
- core/db.cc

5.84 Tempus::Multimodal::Vertex Struct Reference

`#include <multimodal_graph.hh>`

Public Types

- enum **VertexType** { **Road**, **PublicTransport**, **Poi** }

Public Member Functions

- bool **operator==** (const Vertex &v) const
- bool **operator!=** (const Vertex &v) const
- bool **operator<** (const Vertex &v) const
- **Vertex** (const Road::Graph *graph, Road::Vertex vertex)
- **Vertex** (const PublicTransport::Graph *graph, PublicTransport::Vertex vertex)
- **Vertex** (const POI *poi)

Public Attributes

- VertexType **type**
- union {
  const Road::Graph * **road_graph**
  const PublicTransport::Graph * **pt_graph**
  const POI * **poi**
  };
- Road::Vertex **road_vertex**
- PublicTransport::Vertex **pt_vertex**

5.84.1 Detailed Description

A Multimodal::Vertex is either a Road::Vertex or PublicTransport::Vertex on a particular public transport network

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

- core/multimodal_graph.hh
- core/multimodal_graph.cc

5.85 Tempus::vertex_or_edge< G, Tag > Struct Template Reference

Classes

- struct null_class

Public Types

- typedef null_class **property_type**
- typedef null_class **descriptor**

template<class G, class Tag> struct Tempus::vertex_or_edge< G, Tag >

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

- core/multimodal_graph.hh

5.86 Tempus::vertex_or_edge< G, boost::edge_property_tag > Struct Template Reference

Public Types

- typedef boost::edge_bundle_type< G >::type **property_type**
- typedef boost::graph_traits< G >::edge_descriptor **descriptor**

template<class G> struct Tempus::vertex_or_edge< G, boost::edge_property_tag >

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

- core/multimodal_graph.hh

5.87 Tempus::vertex_or_edge< G, boost::vertex_property_tag > Struct Template Reference

Public Types

- typedef boost::vertex_bundle_type< G >::type **property_type**
- typedef boost::graph_traits< G >::vertex_descriptor **descriptor**

template<class G> struct Tempus::vertex_or_edge< G, boost::vertex_property_tag >

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

- core/multimodal_graph.hh

5.88 Tempus::Multimodal::VertexIndexProperty Class Reference

Public Member Functions

- **VertexIndexProperty** (const Graph &graph)
- size_t **get_index** (const Vertex &v) const
- size_t **operator[]** (const Vertex &v) const

Protected Attributes

- const Multimodal::Graph & **graph_**

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

- core/multimodal_graph.hh
- core/multimodal_graph.cc

5.89 Tempus::Multimodal::VertexIterator Struct Reference

Public Member Functions

- **VertexIterator** (const Graph &graph)
- void **to_end** ()
- Vertex & **dereference** () const
- void **increment** ()
- bool **equal** (const VertexIterator &v) const

Protected Attributes

- Road::VertexIterator **road_it_**
- Road::VertexIterator **road_it_end_**
- Multimodal::Graph::PublicTransportGraphList::const_iterator **pt_graph_it_**
- Multimodal::Graph::PublicTransportGraphList::const_iterator **pt_graph_it_end_**
- Multimodal::Graph::PoiList::const_iterator **poi_it_**
- Multimodal::Graph::PoiList::const_iterator **poi_it_end_**
- PublicTransport::VertexIterator **pt_it_**
- PublicTransport::VertexIterator **pt_it_end_**
- const Multimodal::Graph * **graph_**
- Multimodal::Vertex **vertex_**

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

- core/multimodal_graph.hh
- core/multimodal_graph.cc

5.90 wps_client::WPSClient Class Reference

Public Member Functions

- def **__init__**
- def **get_capabilities**
- def **describe_process**
- def **execute**

Public Attributes

- **conn**

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

- wps/client/wps_client.py

5.91 XML Class Reference

`#include <xml_helper.hh>`

Static Public Member Functions

- static std::string escape_text (const std::string &message)
- static std::string to_string (xmlNode ∗node, int indent_level=0)
- static void ensure_validity (xmlNode ∗node, const std::string &schema_str)
- static xmlNode ∗ **new_node** (const std::string &name)
- template<class T >
  static void **new_prop** (xmlNode ∗node, const std::string &key, T value)
- static void **add_child** (xmlNode ∗node, xmlNode ∗child)
- static xmlNode ∗ get_next_nontext (xmlNode ∗node)

Static Protected Member Functions

- static void accumulate_error (void ∗ctx, const char ∗msg,...)
- static int **init** ()

Static Protected Attributes

- static bool **clear_errors_** = false
- static std::string **xml_error_**
- static int **init_n_** = XML::init()

5.91.1 Detailed Description

XML helper class

5.91.2 Member Function Documentation

5.91.2.1 void XML::accumulate_error (void ∗ *ctx*, const char ∗ *msg*, ...) [static, protected]

Generic libxml error handling. Accumulate errors in a string. This is intended to be used to transform XML parsing errors to std::exceptions

5.91.2.2 void XML::ensure_validity (xmlNode ∗ *node*, const std::string & *schema_str*) [static]

Throws a std::invalid_argument if the given node is not validated against the schema

5.91.2.3 std::string XML::escape_text (const std::string & *message*) [static]

Returns a string that can be written as an XML text node

5.91.2.4 xmlNode * XML::get_next_nontext (xmlNode * *node*) [static]

Get the next non text node

5.91.2.5 std::string XML::to_string (xmlNode * *node,* int *indent_level* = 0) [static]

Outputs a node to a string, recursively

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

- wps/xml_helper.hh
- wps/xml_helper.cc