Justine - a rapid prototype for development of GNU Robocar City Emulator and Robocar World Championship

0.0.12

Generated by Doxygen 1.8.6

Tue Mar 10 2015 16:54:19

Contents

1	Just	ine - th	is is a rap	pia p	prototy	/pe to	or a	evel	opn	nent	Of I	Rob	OC	ar C	ity	Emi	ulate	or					1
	1.1	Introdu	iction																 	 	 		1
2	Hier	archica	l Index																				3
	2.1	Class	Hierarchy	٠															 	 	 	-	3
3	Clas	s Index																					5
	3.1	Class	List																 		 	-	5
4	File	Index																					7
	4.1	File Lis	st																 	 	 		7
5	Clas	s Docu	mentatior	n																			9
	5.1	justine	::robocar::	::Ant	Car Cl	ass F	Refer	renc	e.										 	 	 		9
		5.1.1	Detailed	d De	scriptic	on .													 	 	 		9
	5.2	justine	::robocar::	::Car	Class	Refe	erend	се											 	 	 		10
		5.2.1	Detailed	d De	scriptic	on .													 	 	 		10
	5.3	justine	::robocar::	::Car	Lexer	Class	s Re	fere	nce										 	 	 		11
		5.3.1	Detailed	d De	scriptic	on .													 	 	 		11
	5.4	justine	::robocar::	::Co	oCar C	lass	Refe	eren	се										 	 	 		11
		5.4.1	Detailed	d De	scriptic	on .													 	 	 		12
	5.5	justine	::samplecl	client	t::MySł	hmCl	lient	Clas	ss R	efer	ence	. .							 	 	 		12
		5.5.1	Detailed	d De	scriptic	on .													 	 	 		13
		5.5.2	Construc	ictor	& Des	tructo	or Do	ocun	nen	tatio	n								 	 	 		13
			5.5.2.1	M	yShm(Client	it												 	 	 		13
		5.5.3	Member	r Fur	nction I	Docu	ımen	ntatio	on										 	 	 		13
			5.5.3.1	b	gl_grap	oh .													 	 	 		13
			5.5.3.2	ha	asBelln	manF	ordF	Path											 	 	 		14
			5.5.3.3	ha	asDijks	straPa	ath												 	 	 		15
			5.5.3.4	nı	um_vei	rtices	s												 	 	 		15
			5.5.3.5	pr	rint_ed	iges													 	 	 		15
			5.5.3.6	pr	rint_ve	rtices	s												 	 	 		16
			5.5.3.7	st	art .														 		 		16

iv CONTENTS

5.6	justine	::robocar::OSMReader Class Reference	16
	5.6.1	Detailed Description	17
5.7	justine	::robocar::SharedData Class Reference	17
	5.7.1	Detailed Description	17
5.8	justine	::sampleclient::ShmClient Class Reference	17
	5.8.1	Detailed Description	18
	5.8.2	Constructor & Destructor Documentation	19
		5.8.2.1 ShmClient	19
	5.8.3	Member Function Documentation	20
		5.8.3.1 alist	20
		5.8.3.2 get_random_node	20
		5.8.3.3 num_edges	20
		5.8.3.4 start	20
5.9	justine	::robocar::SmartCar Class Reference	21
	5.9.1	Detailed Description	21
5.10	justine	::robocar::SmartCity Class Reference	21
	5.10.1	Detailed Description	22
5.11	justine	::robocar::Traffic Class Reference	22
	5.11.1	Detailed Description	23
5.12	yy_buf	fer_state Struct Reference	23
	5.12.1	Detailed Description	23
	5.12.2	Member Data Documentation	24
		5.12.2.1 yy_bs_column	24
		5.12.2.2 yy_bs_lineno	24
5.13	yy_trar	ns_info Struct Reference	24
	5.13.1	Detailed Description	24
Eilo	Dooum	entation 2	25
6.1			25 25
0.1	6.1.1		-5 25
	6.1.2		25
	6.1.3		<u>2</u> 5
6.2			26
0.2	6.2.1		-0 26
	6.2.2	•	-0 26
	6.2.3		-0 26
6.3			-0 27
0.0	6.3.1		- [,] 27
	6.3.2		- [,] 27
	6.3.3		- 1 27
	0.0.0	ZEOGRIF HON	-1

6

CONTENTS

src/mai	inpage.h File Reference	27
6.4.1	Detailed Description	28
src/mys	shmclient-main.cpp File Reference	28
6.5.1	Detailed Description	28
6.5.2	LICENSE	28
6.5.3	DESCRIPTION	28
src/mys	shmclient.cpp File Reference	29
6.6.1	Detailed Description	29
6.6.2	LICENSE	29
6.6.3	DESCRIPTION	29
src/mys	shmclient.hpp File Reference	29
6.7.1	Detailed Description	31
6.7.2	LICENSE	31
6.7.3	DESCRIPTION	31
src/osn	nreader.hpp File Reference	31
6.8.1	Detailed Description	32
6.8.2	LICENSE	33
6.8.3	DESCRIPTION	33
src/shn	nclient-main.cpp File Reference	33
6.9.1	Detailed Description	33
6.9.2	LICENSE	34
6.9.3	DESCRIPTION	34
src/shn	nclient.cpp File Reference	34
6.10.1	Detailed Description	34
6.10.2	LICENSE	34
6.10.3	DESCRIPTION	35
src/shn	nclient.hpp File Reference	35
6.11.1	Detailed Description	35
6.11.2	LICENSE	35
6.11.3	DESCRIPTION	36
src/sma	artcity.hpp File Reference	36
6.12.1	Detailed Description	37
6.12.2	LICENSE	37
6.12.3	DESCRIPTION	37
src/traf	fic-main.cpp File Reference	37
6.13.1	Detailed Description	38
6.13.2	LICENSE	38
6.13.3	DESCRIPTION	38
src/traf	fic.cpp File Reference	38
6.14.1	Detailed Description	38
	6.4.1 src/mys 6.5.1 6.5.2 6.5.3 src/mys 6.6.1 6.6.2 6.6.3 src/mys 6.7.1 6.7.2 6.7.3 src/osn 6.8.1 6.8.2 6.8.3 src/shn 6.9.1 6.10.2 6.10.3 src/shn 6.11.1 6.11.2 6.11.3 src/shn 6.11.1 6.11.2 6.11.3 src/shn 6.12.1 6.12.2 6.12.3 src/traf 6.13.1 6.13.2 6.13.3 src/traf 6.13.1 6.13.2 6.13.3 src/traf	src/myshmclient-main.cpp File Reference 6.5.1 Detailed Description 6.5.2 LICENSE 6.5.3 DESCRIPTION src/myshmclient.cpp File Reference 6.6.1 Detailed Description 6.6.2 LICENSE 6.6.3 DESCRIPTION src/myshmclient.hpp File Reference 6.7.1 Detailed Description 6.7.2 LICENSE 6.7.3 DESCRIPTION src/osmreader.hpp File Reference 6.8.1 Detailed Description 6.8.2 LICENSE 6.8.3 DESCRIPTION src/osmreader.hpp File Reference 6.8.1 Detailed Description 6.8.2 LICENSE 6.8.3 DESCRIPTION src/shmclient-main.cpp File Reference 6.9.1 Detailed Description 6.9.2 LICENSE

•	CONTENTS
<i>i</i> i	CONTENTS
VI	CONTENTS

6.14.2	LICENSE	 																-	39
6.14.3	DESCRIPTION	 	 																39

Chapter 1

Justine - this is a rapid prototype for development of Robocar City Emulator

Authors

Norbert Bátfai nbatfai@gmail.com

1.1 Introduction

2	Justine - this is a rapid prototype for development of Hobocar City Emulator

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

justine::robocar::Car
justine::robocar::AntCar
justine::robocar::SmartCar
justine::robocar::CopCar
Handler
justine::robocar::OSMReader
justine::robocar::SharedData
justine::sampleclient::ShmClient
justine::sampleclient::MyShmClient
justine::robocar::SmartCity
justine::robocar::Traffic
yy_buffer_state
yy_trans_info
yyFlexLexer
iustine::robocar::CarLexer

Hierarchical Index

Chapter 3

Class Index

3.1 Class List

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

justine::robocar::AntCar	9
justine::robocar::Car	10
justine::robocar::CarLexer	11
justine::robocar::CopCar	11
justine::sampleclient::MyShmClient	
A sample class used for testing the routing algorithms	12
justine::robocar::OSMReader	16
justine::robocar::SharedData	17
justine::sampleclient::ShmClient	
A sample class used for testing IPC mechanisms (SHM and sockets) which are used by the city	
emulator	17
justine::robocar::SmartCar	21
justine::robocar::SmartCity	21
justine::robocar::Traffic	22
yy_buffer_state	23
yy_trans_info	24

6 Class Index

Chapter 4

File Index

4.1 File List

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

src/car.cpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	25
src/car.hpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	26
src/carlexer.cc	??
src/carlexer.hpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	27
src/mainpage.h	
Justine - this is a rapid prototype for development of Robocar City Emulator	27
src/myshmclient-main.cpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	28
src/myshmclient.cpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	29
src/myshmclient.hpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	29
src/osmreader.hpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	31
src/shmclient-main.cpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	33
src/shmclient.cpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	34
src/shmclient.hpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	35
src/smartcity.cpp	??
src/smartcity.hpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	36
src/traffic-main.cpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	37
src/traffic.cpp	
Justine - this is a rapid prototype for development of Robocar City Emulator	38
src/traffic.hpp	??

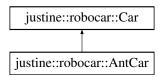
8 File Index

Chapter 5

Class Documentation

5.1 justine::robocar::AntCar Class Reference

Inheritance diagram for justine::robocar::AntCar:



Public Member Functions

- AntCar (Traffic &traffic)
- virtual void nextSmarterEdge (void)
- virtual void print (std::ostream &os) const
- osmium::unsigned_object_id_type ant (void)
- osmium::unsigned_object_id_type ant_rnd (void)
- osmium::unsigned_object_id_type ant_rernd (void)
- osmium::unsigned_object_id_type ant_mrernd (void)

Static Public Attributes

- static AdjacencyList alist
- static AdjacencyList alist_evaporate

Additional Inherited Members

5.1.1 Detailed Description

Definition at line 126 of file car.hpp.

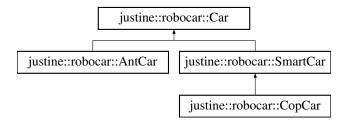
The documentation for this class was generated from the following files:

- src/car.hpp
- · src/car.cpp

10 Class Documentation

5.2 justine::robocar::Car Class Reference

Inheritance diagram for justine::robocar::Car:



Public Member Functions

- Car (Traffic &traffic, CarType type=CarType::NORMAL)
- · virtual void init ()
- virtual void step ()
- osmium::unsigned_object_id_type from () const
- osmium::unsigned_object_id_type to () const
- osmium::unsigned_object_id_type get_step () const
- CarType get_type () const
- void set_type (CarType type)
- osmium::unsigned_object_id_type to_node () const
- osmium::unsigned_object_id_type get_max_steps () const
- virtual void nextEdge (void)
- virtual void nextSmarterEdge (void)
- · virtual void print (std::ostream &os) const

Protected Attributes

- Traffic & traffic
- CarType m_type {CarType::NORMAL}
- osmium::unsigned_object_id_type m_from {3130863972}
- osmium::unsigned_object_id_type m_to {0}
- osmium::unsigned_object_id_type **m_step** {0}

Friends

std::ostream & operator<< (std::ostream &os, Car &c)

5.2.1 Detailed Description

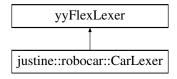
Definition at line 55 of file car.hpp.

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

- src/car.hpp
- · src/car.cpp

5.3 justine::robocar::CarLexer Class Reference

Inheritance diagram for justine::robocar::CarLexer:



Public Member Functions

- virtual int yylex ()
- char * get_name ()
- char **get_role** () const
- int get_num () const
- int get_errnumber () const
- · bool get_guided () const
- int get_cmd () const
- int get_id () const
- std::vector< unsigned int > & get_route (void)
- unsigned int get_from () const
- unsigned int get_to () const

Friends

• std::ostream & operator<< (std::ostream &os, CarLexer &cl)

5.3.1 Detailed Description

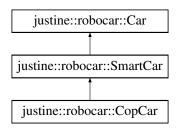
Definition at line 50 of file carlexer.hpp.

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

src/carlexer.hpp

5.4 justine::robocar::CopCar Class Reference

Inheritance diagram for justine::robocar::CopCar:



12 Class Documentation

Public Member Functions

- CopCar (Traffic &traffic, bool guided, const char *name)
- · virtual void print (std::ostream &os) const
- std::string get_name () const
- int get_num_captured_gangsters () const
- void captured_gangster (void)

Protected Attributes

- int m_num_captured_gangsters {0}
- std::string m_name

5.4.1 Detailed Description

Definition at line 203 of file car.hpp.

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

- src/car.hpp
- src/car.cpp

5.5 justine::sampleclient::MyShmClient Class Reference

A sample class used for testing the routing algorithms.

```
#include <myshmclient.hpp>
```

Inheritance diagram for justine::sampleclient::MyShmClient:



Public Member Functions

• MyShmClient (const char *shm segment, std::string teamname)

This constructor creates the BGL graph from the map graph.

• ∼MyShmClient ()

Dtor.

void start (boost::asio::io service &io service, const char *port)

This function starts the client.

- void start10 (boost::asio::io_service &io_service, const char *port)
- int num_vertices (int &sum_edges)

This function counts the number of vertices and number of edges in the map graph.

void print_edges (unsigned more)

This function prints the edges of the map graph.

void print_vertices (unsigned more)

This function prints the vertices of the map graph.

NodeRefGraph * bgl_graph (void)

This function create the BGL graph.

- · std::vector
 - < osmium::unsigned_object_id_type > hasDijkstraPath (osmium::unsigned_object_id_type from, osmium::unsigned_object_id_type to)

This function solves the shortest path problem using Dijkstra algorithm.

- · std::vector
 - < osmium::unsigned_object_id_type > hasBellmanFordPath (osmium::unsigned_object_id_type from, osmium::unsigned_object_id_type to)

This function solves the shortest path problem using Bellman-Ford algorithm.

Protected Attributes

- NodeRefGraph * nr_graph
- std::string m_teamname

5.5.1 Detailed Description

A sample class used for testing the routing algorithms.

This sample class shows how client agents can create BGL graph from data can be found in the shared memory.

Author

Norbert Bátfai

Date

Dec. 7, 2014

Definition at line 105 of file myshmclient.hpp.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 justine::sampleclient::MyShmClient(:const char * shm_segment, std::string teamname) [inline]

This constructor creates the BGL graph from the map graph.

Parameters

```
shm segment the shared memory object name
```

This constructor creates the BGL graph from the map graph that is placed in the shared memory segment.

Definition at line 116 of file myshmclient.hpp.

5.5.3 Member Function Documentation

5.5.3.1 NodeRefGraph* justine::sampleclient::MyShmClient::bgl_graph(void) [inline]

This function create the BGL graph.

Returns

he pointer of the created BGL graph.

Definition at line 249 of file myshmclient.hpp.

14 Class Documentation

This function solves the shortest path problem using Bellman-Ford algorithm.

Parameters

source	the source node
target	the target node

Returns

the shortest path between nodes source and target

This function determines the shortest path from the source node to the target node.

Definition at line 400 of file myshmclient.hpp.

5.5.3.3 std::vector<osmium::unsigned_object_id_type> justine::sampleclient::MyShmClient::hasDijkstraPath (osmium::unsigned_object_id_type from, osmium::unsigned_object_id_type to) [inline]

This function solves the shortest path problem using Dijkstra algorithm.

Parameters

source	the source node
target	the target node

Returns

the shortest path between nodes source and target

This function determines the shortest path from the source node to the target node.

Definition at line 329 of file myshmclient.hpp.

5.5.3.4 int justine::sampleclient::MyShmClient::num_vertices (int & sum_edges) [inline]

This function counts the number of vertices and number of edges in the map graph.

Parameters

out	sum_edges	the number of edges

Returns

the number of vertices

This function counts the number of vertices and number of edges in the map graph that is placed in the shared memory segment.

Definition at line 162 of file myshmclient.hpp.

5.5.3.5 void justine::sampleclient::MyShmClient::print_edges (unsigned *more*) [inline]

This function prints the edges of the map graph.

Parameters

nore	the maximum number of printed items

Definition at line 189 of file myshmclient.hpp.

5.5.3.6 void justine::sampleclient::MyShmClient::print_vertices (unsigned more) [inline]

This function prints the vertices of the map graph.

16 Class Documentation

Parameters

more	the maximum number of printed items

Definition at line 214 of file myshmclient.hpp.

5.5.3.7 void justine::sampleclient::MyShmClient::start (boost::asio::io_service & io_service, const char * port)

This function starts the client.

Parameters

io_service	
port	the TCP port of the traffic server

This method does the following: retrieves a value from shared memory, then establishes a connection with the traffic server, finally sends some client commands.

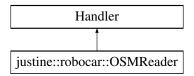
Definition at line 265 of file myshmclient.cpp.

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

- src/myshmclient.hpp
- · src/myshmclient.cpp

5.6 justine::robocar::OSMReader Class Reference

Inheritance diagram for justine::robocar::OSMReader:



Public Member Functions

- OSMReader (const char *osm_file, AdjacencyList &alist, AdjacencyList &palist, WaynodeLocations &waynode_locations, WayNodesMap &busWayNodesMap, Way2Nodes &way2nodes)
- std::size_t get_estimated_memory () const
- bool edge (osmium::unsigned_object_id_type v1, osmium::unsigned_object_id_type v2)
- void **node** (osmium::Node &node)
- void way (osmium::Way &way)
- void relation (osmium::Relation &rel)

Public Attributes

- int onewayc {0}
- int onewayf {false}

Protected Attributes

- Vertices vert
- int nOSM nodes {0}
- int nOSM_ways {0}

- int nOSM_relations {0}
- int sum_unique_highhway_nodes {0}
- int sum_highhway_nodes {0}
- int sum_highhway_length {0}
- int edge_multiplicity = 0
- int nbuses {0}
- double max edge length {0.0}
- double mean_edge_length {0.0}
- int cedges {0}
- · OSMLocations locations

5.6.1 Detailed Description

Definition at line 72 of file osmreader.hpp.

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

· src/osmreader.hpp

5.7 justine::robocar::SharedData Class Reference

Public Member Functions

• SharedData (const void_allocator &void_alloc)

Public Attributes

- uint_vector m_alist
- · uint vector m salist
- uint vector m palist
- int lon
- int lat

5.7.1 Detailed Description

Definition at line 62 of file smartcity.hpp.

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

• src/smartcity.hpp

5.8 justine::sampleclient::ShmClient Class Reference

A sample class used for testing IPC mechanisms (SHM and sockets) which are used by the city emulator.

#include <shmclient.hpp>

Inheritance diagram for justine::sampleclient::ShmClient:



18 Class Documentation

Public Member Functions

ShmClient (const char *shm segment)

This constructor initializes the shared memory segment.

void start (boost::asio::io service &io service, const char *port)

This function starts the client.

virtual

osmium::unsigned_object_id_type get_random_node (void)

This function returns a randomly chosen node from the map.

• size_t num_edges (osmium::unsigned_object_id_type from) const

Returns the number of out edges of a given vertex.

• osmium::unsigned_object_id_type alist (osmium::unsigned_object_id_type from, int to) const

Returns the i-th neighbor of the actual vertex.

- int alist inv (osmium::unsigned object id type from, osmium::unsigned object id type to) const
- osmium::unsigned_object_id_type salist (osmium::unsigned_object_id_type from, int to) const
- void set_salist (osmium::unsigned_object_id_type from, int to, osmium::unsigned_object_id_type value)
- osmium::unsigned object id type palist (osmium::unsigned object id type from, int to) const
- bool **hasNode** (osmium::unsigned_object_id_type node)
- double dst (osmium::unsigned_object_id_type n1, osmium::unsigned_object_id_type n2) const
- double dst (double lon1, double lat1, double lon2, double lat2) const
- void toGPS (osmium::unsigned_object_id_type from, double *lo, double *la) const
- void **toGPS** (osmium::unsigned_object_id_type from, osmium::unsigned_object_id_type to, osmium::unsigned_object_id_type step, double *la) const

Protected Attributes

```
    boost::interprocess::offset_ptr
    justine::robocar::shm map Type > shm map
```

The OSM map data stored in a shared memory segment.

5.8.1 Detailed Description

A sample class used for testing IPC mechanisms (SHM and sockets) which are used by the city emulator.

This sample class shows how client agents can connect and communicate with traffic emulator using shared memory.

Author

Norbert Bátfai

Date

Dec. 7, 2014

Definition at line 66 of file shmclient.hpp.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 justine::sampleclient::ShmClient (const char * shm_segment) [inline]

This constructor initializes the shared memory segment.

Parameters

shm_segment	the shared memory object name
-------------	-------------------------------

This constructor attaches the shared memory segment identified by the param shm_segment.

Definition at line 76 of file shmclient.hpp.

5.8.3 Member Function Documentation

5.8.3.1 osmium::unsigned_object_id_type justine::sampleclient::ShmClient::alist (osmium::unsigned_object_id_type from, int to) const [inline]

Returns the i-th neighbor of the actual vertex.

Parameters

to	the index i

Returns

the (osmium) reference number of the i-th neighbor

This method returns the i-th neighbor of the actual vertex in the shared adjacency list.

Definition at line 141 of file shmclient.hpp.

5.8.3.2 virtual osmium::unsigned_object_id_type justine::sampleclient::ShmClient::get_random_node (void) [inline], [virtual]

This function returns a randomly chosen node from the map.

Returns

the randomly chosen node

This method may be useful if you want to add a new car to the map.

Definition at line 110 of file shmclient.hpp.

5.8.3.3 size_t justine::sampleclient::ShmClient::num_edges (osmium::unsigned_object_id_type from) const [inline]

Returns the number of out edges of a given vertex.

Parameters

from a given	given vertex
--------------	--------------

Returns

the number of edges

This method returns the size of the vector of neighboring vertices in the shared adjacency list.

Definition at line 126 of file shmclient.hpp.

5.8.3.4 void justine::sampleclient::ShmClient::start (boost::asio::io_service & io_service, const char * port)

This function starts the client.

20 Class Documentation

Parameters

io_service	
port	the TCP port of the traffic server

This method does the following: retrieves a value from shared memory, then establishes a connection with the traffic server, finally sends some client commands.

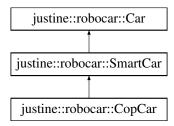
Definition at line 233 of file shmclient.cpp.

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

- src/shmclient.hpp
- · src/shmclient.cpp

5.9 justine::robocar::SmartCar Class Reference

Inheritance diagram for justine::robocar::SmartCar:



Public Member Functions

- SmartCar (Traffic &traffic, CarType type, bool guided)
- · virtual void step ()
- virtual void init ()
- virtual void print (std::ostream &os) const
- · bool get guided () const
- bool **set_route** (std::vector< unsigned int > &route)
- virtual void nextEdge (void)
- virtual void nextGuidedEdge (void)
- bool **set_fromto** (unsigned int from, unsigned int to)

Additional Inherited Members

5.9.1 Detailed Description

Definition at line 163 of file car.hpp.

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

- src/car.hpp
- src/car.cpp

5.10 justine::robocar::SmartCity Class Reference

Public Member Functions

• SmartCity (const char *osm_file, const char *shm_segment, const char *map_file)

- SmartCity (const char *osm_file, const char *shm_segment)
- void processes ()
- · virtual void city_run ()
- double busWayLength (bool verbose)

This function gives a list of all the bus services operating in a given city.

Protected Attributes

- boost::interprocess::managed_shared_memory * segment
- boost::interprocess::offset_ptr< shm_map_Type > shm_map
- int m_delay {5000}
- · bool m_run {true}

Friends

std::ostream & operator<< (std::ostream &os, SmartCity &t)

5.10.1 Detailed Description

Definition at line 83 of file smartcity.hpp.

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

- · src/smartcity.hpp
- · src/smartcity.cpp

5.11 justine::robocar::Traffic Class Reference

Public Member Functions

- Traffic (int size, const char *shm_segment, double catchdist, TrafficType type=TrafficType::NORMAL, int minutes=10)
- void processes ()
- std::string get_title (std::string name)
- osmium::unsigned_object_id_type **node** ()
- virtual void traffic run ()
- · void steps ()
- void pursuit (void)
- size_t nedges (osmium::unsigned_object_id_type from) const
- osmium::unsigned_object_id_type alist (osmium::unsigned_object_id_type from, int to) const
- int alist_inv (osmium::unsigned_object_id_type from, osmium::unsigned_object_id_type to) const
- osmium::unsigned object id type salist (osmium::unsigned object id type from, int to) const
- void set_salist (osmium::unsigned_object_id_type from, int to, osmium::unsigned_object_id_type value)
- · osmium::unsigned object id type palist (osmium::unsigned object id type from, int to) const
- bool hasNode (osmium::unsigned_object_id_type node)
- void start_server (boost::asio::io_service &io_service, unsigned short port)
- void cmd_session (boost::asio::ip::tcp::socket sock)
- osmium::unsigned_object_id_type **naive_node_for_nearest_gangster** (osmium::unsigned_object_id_type from, osmium::unsigned_object_id_type to, osmium::unsigned_object_id_type step)
- double **dst** (osmium::unsigned_object_id_type n1, osmium::unsigned_object_id_type n2) const

22 Class Documentation

- double dst (double lon1, double lat1, double lon2, double lat2) const
- void **toGPS** (osmium::unsigned_object_id_type from, osmium::unsigned_object_id_type to, osmium::unsigned_object_id_type step, double *lo, double *la) const
- TrafficType get_type () const
- int get_time () const

Protected Attributes

- boost::interprocess::managed shared memory * segment
- boost::interprocess::offset_ptr< shm_map_Type > shm_map
- int m delay {200}
- · bool m_run {true}
- double m_catchdist {15.5}

Friends

std::ostream & operator<< (std::ostream &os, Traffic &t)

5.11.1 Detailed Description

Definition at line 81 of file traffic.hpp.

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

- · src/traffic.hpp
- · src/traffic.cpp

5.12 yy_buffer_state Struct Reference

Public Attributes

- std::istream * yy_input_file
- char * yy_ch_buf
- char * yy_buf_pos
- yy_size_t yy_buf_size
- int yy_n_chars
- int yy_is_our_buffer
- int yy_is_interactive
- int yy at bol
- int yy_bs_lineno
- int yy_bs_column
- · int yy fill buffer
- int yy_buffer_status

5.12.1 Detailed Description

Definition at line 208 of file carlexer.cc.

5.12.2 Member Data Documentation

5.12.2.1 int yy_buffer_state::yy_bs_column

The column count.

Definition at line 246 of file carlexer.cc.

5.12.2.2 int yy_buffer_state::yy_bs_lineno

The line count.

Definition at line 245 of file carlexer.cc.

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

· src/carlexer.cc

5.13 yy_trans_info Struct Reference

Public Attributes

- · flex_int32_t yy_verify
- flex_int32_t yy_nxt

5.13.1 Detailed Description

Definition at line 341 of file carlexer.cc.

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

• src/carlexer.cc

24 **Class Documentation**

Chapter 6

File Documentation

6.1 src/car.cpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

```
#include <car.hpp>
#include <traffic.hpp>
#include <boost/iterator/iterator_concepts.hpp>
```

6.1.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

Norbert Bátfai nbatfai@gmail.com

Version

0.0.10

6.1.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.1.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file car.cpp.

26 File Documentation

6.2 src/car.hpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

```
#include <osmium/osm/types.hpp>
#include <iostream>
#include <vector>
#include <osmreader.hpp>
#include <algorithm>
```

Classes

· class justine::robocar::Car

· class justine::robocar::AntCar

· class justine::robocar::SmartCar

· class justine::robocar::CopCar

Enumerations

enum CarType : unsigned int { NORMAL =0, POLICE, GANGSTER, CAUGHT }

6.2.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

Norbert Bátfai nbatfai@gmail.com

Version

0.0.10

6.2.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.2.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file car.hpp.

6.3 src/carlexer.hpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

```
#include <FlexLexer.h>
#include <iostream>
#include <sstream>
#include <cstring>
#include <cstdio>
#include <vector>
```

Classes

· class justine::robocar::CarLexer

6.3.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

```
Norbert Bátfai nbatfai@gmail.com
```

Version

0.0.10

6.3.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.3.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file carlexer.hpp.

6.4 src/mainpage.h File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

28 File Documentation

6.4.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

```
Norbert Bátfai nbatfai@gmail.com
```

Definition in file mainpage.h.

6.5 src/myshmclient-main.cpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

```
#include <myshmclient.hpp>
#include <boost/program_options.hpp>
```

Functions

• int main (int argc, char *argv[])

6.5.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

```
Norbert Bátfai nbatfai@gmail.com
```

Version

0.0.10

6.5.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.5.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file myshmclient-main.cpp.

6.6 src/myshmclient.cpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

#include <myshmclient.hpp>

Variables

· char data [524288]

6.6.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

Norbert Bátfai nbatfai@gmail.com

Version

0.0.10

6.6.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.6.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file myshmclient.cpp.

6.7 src/myshmclient.hpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

30 File Documentation

```
#include <boost/interprocess/managed_shared_memory.hpp>
#include <boost/interprocess/allocators/allocator.hpp>
#include <boost/interprocess/containers/map.hpp>
#include <boost/interprocess/containers/vector.hpp>
#include <boost/interprocess/containers/string.hpp>
#include <smartcity.hpp>
#include <car.hpp>
#include <cstdlib>
#include <iterator>
#include <boost/asio.hpp>
#include <limits>
#include <memory>
#include <boost/graph/adjacency_list.hpp>
#include <boost/graph/graph_traits.hpp>
#include <boost/graph/dijkstra_shortest_paths.hpp>
#include <boost/graph/properties.hpp>
#include <boost/property_map/property_map.hpp>
#include <shmclient.hpp>
#include <algorithm>
#include <boost/graph/bellman_ford_shortest_paths.hpp>
#include <boost/graph/graphviz.hpp>
#include <fstream>
```

Classes

· class justine::sampleclient::MyShmClient

A sample class used for testing the routing algorithms.

Typedefs

```
· typedef boost::adjacency list
 < boost::listS, boost::vecS,
 boost::directedS,
 boost::property
  < boost::vertex_name_t,
 osmium::unsigned object id type >
 , boost::property
  < boost::edge_weight_t, int > > justine::sampleclient::NodeRefGraph
· typedef boost::graph_traits
  < NodeRefGraph >
 ::vertex descriptor justine::sampleclient::NRGVertex
· typedef boost::graph traits
  < NodeRefGraph >
 ::vertex_iterator justine::sampleclient::NRGVertexIter
typedef boost::graph_traits
  < NodeRefGraph >
 ::edge_descriptor justine::sampleclient::NRGEdge
typedef boost::graph_traits
  < NodeRefGraph >
 ::edge iterator justine::sampleclient::NRGEdgelter
· typedef boost::property map
  < NodeRefGraph,
 boost::vertex name t >::type justine::sampleclient::VertexNameMap
· typedef boost::property map
  < NodeRefGraph,
```

boost::vertex_index_t >::type justine::sampleclient::VertexIndexMap

· typedef

boost::iterator_property_map

< NRGVertex *, VertexIndexMap,

NRGVertex, NRGVertex & > justine::sampleclient::PredecessorMap

typedef

boost::iterator_property_map

< int *, VertexIndexMap, int,

int & > justine::sampleclient::DistanceMap

· typedef boost::property_map

< NodeRefGraph,

boost::edge_weight_t >::type justine::sampleclient::EdgeWeightMap

6.7.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

Norbert Bátfai nbatfai@gmail.com

Version

0.0.10

6.7.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.7.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file myshmclient.hpp.

6.8 src/osmreader.hpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

32 File Documentation

```
#include <osmium/io/any_input.hpp>
#include <osmium/handler.hpp>
#include <osmium/visitor.hpp>
#include <osmium/osm/node.hpp>
#include <osmium/osm/way.hpp>
#include <osmium/osm/relation.hpp>
#include <osmium/index/map/sparse_mem_table.hpp>
#include <osmium/index/map/sparse_mem_map.hpp>
#include <osmium/handler/node_locations_for_ways.hpp>
#include <osmium/geom/haversine.hpp>
#include <osmium/geom/coordinates.hpp>
#include <iostream>
#include <map>
#include <set>
#include <vector>
#include <string>
#include <algorithm>
#include <fstream>
#include <exception>
#include <stdexcept>
```

Classes

· class justine::robocar::OSMReader

Typedefs

· typedef

```
osmium::index::map::SparseMemMap
 < osmium::unsigned_object_id_type,
 osmium::Location > justine::robocar::OSMLocations
· typedef std::vector
  < osmium::unsigned object id type > justine::robocar::WayNodesVect

    typedef std::map< std::string,</li>

 WayNodesVect > justine::robocar::WayNodesMap

    typedef std::map

  < osmium::unsigned_object_id_type,
 osmium::Location > justine::robocar::WaynodeLocations

    typedef std::map

  < osmium::unsigned_object_id_type,
 WayNodesVect > justine::robocar::Way2Nodes
· typedef std::map
  < osmium::unsigned object id type,
 WayNodesVect > justine::robocar::AdjacencyList
· typedef
 osmium::index::map::SparseMemMap
 < osmium::unsigned object id type,
 int > justine::robocar::Vertices
```

6.8.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

Norbert Bátfai nbatfai@gmail.com

Version

0.0.10

6.8.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.8.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file osmreader.hpp.

6.9 src/shmclient-main.cpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

```
#include <shmclient.hpp>
#include <boost/program_options.hpp>
```

Functions

• int main (int argc, char *argv[])

6.9.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

```
Norbert Bátfai nbatfai@gmail.com
```

Version

0.0.10

34 File Documentation

6.9.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.9.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file shmclient-main.cpp.

6.10 src/shmclient.cpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

```
#include <shmclient.hpp>
```

Variables

• char data [524288]

6.10.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

Norbert Bátfai nbatfai@gmail.com

Version

0.0.10

6.10.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.10.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file shmclient.cpp.

6.11 src/shmclient.hpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

```
#include <boost/interprocess/managed_shared_memory.hpp>
#include <boost/interprocess/containers/map.hpp>
#include <boost/interprocess/containers/vector.hpp>
#include <boost/interprocess/containers/vector.hpp>
#include <boost/interprocess/containers/string.hpp>
#include <smartcity.hpp>
#include <car.hpp>
#include <cstdlib>
#include <iterator>
#include <boost/asio.hpp>
#include <limits>
#include <memory>
```

Classes

· class justine::sampleclient::ShmClient

A sample class used for testing IPC mechanisms (SHM and sockets) which are used by the city emulator.

6.11.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

Norbert Bátfai nbatfai@gmail.com

Version

0.0.10

6.11.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

36 File Documentation

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.11.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file shmclient.hpp.

6.12 src/smartcity.hpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

```
#include <osmreader.hpp>
#include <thread>
#include <chrono>
#include <condition_variable>
#include <mutex>
#include <boost/interprocess/managed_shared_memory.hpp>
#include <boost/interprocess/allocators/allocator.hpp>
#include <boost/interprocess/containers/map.hpp>
#include <boost/interprocess/containers/vector.hpp>
#include <boost/interprocess/containers/string.hpp>
#include <boost/interprocess/containers/string.hpp>
#include <stdexception>
#include <iomanip>
```

Classes

- · class justine::robocar::SharedData
- · class justine::robocar::SmartCity

Typedefs

· typedef

boost::interprocess::managed_shared_memory::segment_manager justine::robocar::segment_manager_ _Type

· typedef

boost::interprocess::allocator

< void, segment_manager_Type > justine::robocar::void_allocator

· typedef

boost::interprocess::allocator

< unsigned int.

segment_manager_Type > justine::robocar::uint_allocator

typedef

boost::interprocess::vector

< unsigned int, uint_allocator > justine::robocar::uint_vector

typedef
 boost::interprocess::allocator
 < uint_vector,
 segment_manager_Type > justine::robocar::uint_vector_allocator
 typedef std::pair < const
 unsigned int, SharedData > justine::robocar::map_pair_Type
 typedef
 boost::interprocess::allocator
 < map_pair_Type,
 segment_manager_Type > justine::robocar::map_pair_Type_allocator
 typedef
 boost::interprocess::map
 < unsigned int, SharedData,
 std::less < unsigned int >
 , map_pair_Type_allocator > justine::robocar::shm_map_Type

6.12.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

Norbert Bátfai nbatfai@gmail.com

Version

0.0.10

6.12.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.12.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file smartcity.hpp.

6.13 src/traffic-main.cpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

```
#include <traffic.hpp>
#include <boost/program_options.hpp>
```

38 File Documentation

Functions

• int main (int argc, char *argv[])

6.13.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

Norbert Bátfai nbatfai@gmail.com

Version

0.0.10

6.13.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.13.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file traffic-main.cpp.

6.14 src/traffic.cpp File Reference

Justine - this is a rapid prototype for development of Robocar City Emulator.

```
#include <traffic.hpp>
```

6.14.1 Detailed Description

Justine - this is a rapid prototype for development of Robocar City Emulator.

Author

Norbert Bátfai nbatfai@gmail.com

Version

0.0.10

6.14.2 LICENSE

Copyright (C) 2014 Norbert Bátfai, batfai.norbert@inf.unideb.hu

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http-://www.gnu.org/licenses/.

6.14.3 DESCRIPTION

Robocar City Emulator and Robocar World Championship

desc

Definition in file traffic.cpp.

Index

justine::sampleclient::MyShmClient, 13 get_random_node justine::sampleclient::ShmClient, 20 hasBellmanFordPath justine::sampleclient::MyShmClient, 13 hasDijkstraPath justine::sampleclient::MyShmClient, 15 justine::robocar::AntCar, 9 justine::robocar::Car, 10 justine::robocar::Car, 10 justine::robocar::CopCar, 11 justine::robocar::SmartCar, 21 justine::robocar::SmartCar, 21 justine::robocar::SmartCar, 21 justine::robocar::SmartCar, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::robocar::Taffic, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 16 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	alist	ShmClient
justine::sampleclient::MyShmClient, 13 get_random_node justine::sampleclient::ShmClient, 20 hasBellmanFordPath justine::sampleclient::MyShmClient, 13 hasDijkstraPath justine::robocar::AntCar, 9 justine::robocar::Car, 10 justine::robocar::Car, 10 justine::robocar::SharedData, 17 justine::robocar::SharedData, 17 justine::robocar::SharedData, 17 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::robocar::Taffic, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasBellmanFordPath, 13 hasBellmanFordPath, 13 num_vertices, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	justine::sampleclient::ShmClient, 20	justine::sampleclient::ShmClient, 19
justine::sampleclient::MyShmClient, 13 get_random_node justine::sampleclient::ShmClient, 20 hasBellmanFordPath justine::sampleclient::MyShmClient, 13 hasDijkstraPath justine::sampleclient::MyShmClient, 15 justine::robocar::AntCar, 9 justine::robocar::Car, 10 justine::robocar::Car, 10 justine::robocar::CopCar, 11 justine::robocar::SmartCar, 21 justine::robocar::SmartCar, 21 justine::robocar::SmartCar, 21 justine::robocar::SmartCar, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::robocar::Taffic, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 16 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15		src/car.cpp, 25
get_random_node justine::sampleclient::ShmClient, 20 hasBellmanFordPath justine::sampleclient::MyShmClient, 13 hasDijkstraPath justine::sampleclient::MyShmClient, 15 justine::robocar::AntCar, 9 justine::robocar::CarLexer, 11 justine::robocar::CarCopCar, 11 justine::robocar::SharardData, 17 justine::robocar::SmartCar, 21 justine::robocar::SmartCity, 21 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasBellmanFordPath, 13 hasBellmanFordPath, 13 hasBellmanFordPath, 15 MyShmClient, 19 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	bgl_graph	src/car.hpp, 26
get_random_node justine::sampleclient::ShmClient, 20 hasBellmanFordPath justine::sampleclient::MyShmClient, 13 hasDijkstraPath justine::sampleclient::MyShmClient, 15 justine::robocar::AntCar, 9 justine::robocar::Cart_exer, 11 justine::robocar::CopCar, 11 justine::robocar::SharedData, 17 justine::robocar::SmartCar, 21 justine::robocar::SmartCar, 21 justine::robocar::SmartCar, 21 justine::robocar::SmartCity, 21 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	justine::sampleclient::MyShmClient, 13	src/carlexer.hpp, 27
justine::sampleclient::MyShmClient, 13 hasbilkstraPath justine::sampleclient::MyShmClient, 13 hasbilkstraPath justine::sampleclient::MyShmClient, 15 justine::robocar::AntCar, 9 justine::robocar::CarLexer, 11 justine::robocar::CarLexer, 11 justine::robocar::SharedData, 17 justine::robocar::SharedData, 17 justine::robocar::SmartCar, 21 justine::robocar::SmartCity, 21 justine::robocar::Traffic, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasbilkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15		src/mainpage.h, 27
hasBellmanFordPath		src/myshmclient-main.cpp, 28
ijustine::sampleclient::MyShmClient, 13 hasDijkstraPath justine::sampleclient::MyShmClient, 15 justine::robocar::AntCar, 9 justine::robocar::Car, 10 justine::robocar::Car, 11 justine::robocar::Car, 11 justine::robocar::Car, 12 justine::robocar::SmartCar, 21 justine::robocar::SmartCar, 21 justine::robocar::SmartCly, 21 justine::robocar::SmartCly, 21 justine::robocar::SmartCly, 21 justine::robocar::Taffic, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	justine::sampleclient::ShmClient, 20	src/myshmclient.cpp, 29
justine::sampleclient::MyShmClient, 13 hasDijkstraPath justine::robocar::AntCar, 9 justine::robocar::Cart.exer, 11 justine::robocar::CopCar, 10 justine::robocar::CopCar, 11 justine::robocar::ShardCar, 21 justine::robocar::SmartCar, 21 justine::robocar::SmartCar, 21 justine::robocar::Tarffic, 22 justine::robocar::Tarffic, 22 justine::robocar::Tarffic, 22 justine::robocar::Tarffic, 22 justine::robocar::Tarffic, 22 justine::robocar::Tarffic, 23 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	h D - Illon - o F - o - ID - Ale	src/myshmclient.hpp, 29
justine::sampleclient::MyShmClient, 13 hasDijkstraPath justine::robocar::Car(2, 2) justine::robocar::Car(3, 10) justine::robocar::SharedData, 17 justine::robocar::SmartCar(3, 21) justine::robocar::SmartCar(3, 21) justine::robocar::SmartCar(3, 21) justine::robocar::SmartCar(3, 21) justine::robocar::SmartCar(3, 21) justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15		src/osmreader.hpp, 31
ijustine::sampleclient::MyShmClient, 15 justine::robocar::AntCar, 9 justine::robocar::Cart_exer, 11 justine::robocar::Cart_exer, 11 justine::robocar::Cart_exer, 11 justine::robocar::Cart_exer, 11 justine::robocar::Cart_exer, 11 justine::robocar::SharedData, 17 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15		• •
justine::sampleclient::MyShmClient, 15 justine::robocar::Art (2r, 9) justine::robocar::Car, 10 justine::robocar::Car, 21 justine::robocar::CopCar, 11 justine::robocar::SharedData, 17 justine::robocar::SharedData, 17 justine::robocar::SmartCar, 21 justine::robocar::SmartCity, 21 justine::robocar::Traffic, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	-	• •
justine::robocar::AntCar, 9 justine::robocar::Car, 10 justine::robocar::CopCar, 11 justine::robocar::CopCar, 11 justine::robocar::CopCar, 11 justine::robocar::SharedData, 17 justine::robocar::SmartCar, 21 justine::robocar::SmartCar, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasBellmanFordPath, 13 hasBellmanFordPath, 13 num_vertices, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	justine::sampleclient::MyShmClient, 15	• •
justine::robocar::Car, 10 justine::robocar::CopCar, 11 justine::robocar::CopCar, 11 justine::robocar::SharedData, 17 justine::robocar::SharedData, 17 justine::robocar::SmartCity, 21 justine::robocar::Traffic, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasBilmanFordPath, 13 num_vertices, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::ShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15		
justine::robocar::CarLexer, 11 justine::robocar::CopCar, 11 justine::robocar::SharedData, 17 justine::robocar::SharedData, 17 justine::robocar::SharedData, 17 justine::robocar::SmartCar, 21 justine::robocar::SmartCity, 21 justine::robocar::Traffic, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_edges, 15 print_etrices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 MyShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	-	
justine::robocar::CopCar, 11 justine::robocar::CopCar, 11 justine::robocar::SharedData, 17 justine::robocar::SmartCar, 21 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasBellmanFordPath, 13 num_vertices, 15 print_edges, 15 print_edges, 15 print_edges, 15 print_edges, 15 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	-	• •
justine::robocar::OspCart, 11 justine::robocar::OspCart, 11 justine::robocar::OspCart, 11 justine::robocar::SharedData, 17 justine::robocar::SmartCar, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasBellmanFordPath, 13 hasBijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	-	
justine::robocar::SharedData, 17 justine::robocar::SmartCar, 21 justine::robocar::SmartCity, 21 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_edges justine::sampleclient::MyShmClient, 15 mum_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	-	
justine::robocar::SmartCar, 21 justine::robocar::SmartCity, 21 justine::robocar::SmartCity, 21 justine::robocar::Traffic, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15		
justine::robocar::SmartCity, 21 justine::robocar::Traffic, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	-	justinosampiosiisritsrimoiisrit, 20
justine::robocar::Traffic, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 mum_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	-	vy bs column
justine::robocar::rramc, 22 justine::sampleclient::MyShmClient, 12 bgl_graph, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_edges, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	·	• •
bgl_graph, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::MyShmClient, 13 mum_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	-	
bgl_graph, 13 hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15		• • •
hasBellmanFordPath, 13 hasDijkstraPath, 15 MyShmClient, 13 num_vertices, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	·	
MyShmClient, 13 num_edges justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 15 my_bs_lineno, 24 yy_bs_lineno, 24 yy_trans_info, 24 yy_bs_lineno, 24 ylellineno, 24 ylellineno, 24 ylellineno, 24 ylellineno, 24 yl	hasBellmanFordPath, 13	
mySnmClient, 13 num_vertices, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	hasDijkstraPath, 15	
num_vertices, 15 print_edges, 15 print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	MyShmClient, 13	
print_vertices, 16 start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	num_vertices, 15	yy_trans_ino, 24
start, 16 justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	print_edges, 15	
justine::sampleclient::ShmClient, 17 alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	print_vertices, 16	
alist, 20 get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	start, 16	
get_random_node, 20 num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	justine::sampleclient::ShmClient, 17	
num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	alist, 20	
num_edges, 20 ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	get_random_node, 20	
ShmClient, 19 start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15		
start, 20 MyShmClient justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	_ ·	
justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15		
justine::sampleclient::MyShmClient, 13 num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	MvShmClient	
num_edges justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	-	
justine::sampleclient::ShmClient, 20 num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	justinesampieciientiwyoninolient, 10	
num_vertices justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	num_edges	
justine::sampleclient::MyShmClient, 15 print_edges justine::sampleclient::MyShmClient, 15	justine::sampleclient::ShmClient, 20	
print_edges justine::sampleclient::MyShmClient, 15	num_vertices	
justine::sampleclient::MyShmClient, 15	justine::sampleclient::MyShmClient, 15	
justine::sampleclient::MyShmClient, 15	print edges	
	. – •	
	print_vertices	
justine::sampleclient::MyShmClient, 16	• —	