

CrowdSenSim

1.1

Generated by Doxygen 1.8.11

Contents

1	Class Index	1
1.1	Class List	1
2	File Index	3
2.1	File List	3
3	Class Documentation	5
3.1	Antenna Struct Reference	5
3.1.1	Constructor & Destructor Documentation	5
3.1.1.1	Antenna(int A, string B, Location C)	5
3.1.2	Member Data Documentation	5
3.1.2.1	id_antenna	5
3.1.2.2	loc	5
3.1.2.3	type_antenna	6
3.2	Event Struct Reference	6
3.2.1	Constructor & Destructor Documentation	6
3.2.1.1	Event(int A, Location B, tm C)	6
3.2.2	Member Data Documentation	6
3.2.2.1	id_user	6
3.2.2.2	loc	6
3.2.2.3	timestamp	6
3.3	Location Struct Reference	7
3.3.1	Constructor & Destructor Documentation	7
3.3.1.1	Location(float A, float B, float C)	7

3.3.2	Member Data Documentation	7
3.3.2.1	alt	7
3.3.2.2	lat	7
3.3.2.3	lon	7
3.4	LocationA Struct Reference	7
3.4.1	Constructor & Destructor Documentation	8
3.4.1.1	LocationA(int A, double B, double C, double D)	8
3.4.2	Member Data Documentation	8
3.4.2.1	alt	8
3.4.2.2	id	8
3.4.2.3	lat	8
3.4.2.4	lon	8
3.5	Position Struct Reference	8
3.5.1	Constructor & Destructor Documentation	8
3.5.1.1	Position(Location A, tm B)	8
3.5.2	Member Data Documentation	8
3.5.2.1	loc	8
3.5.2.2	timestamp	9
3.6	Sample Struct Reference	9
3.6.1	Constructor & Destructor Documentation	9
3.6.1.1	Sample(string A, float B, float C, bool D, Position E)	9
3.6.2	Member Data Documentation	9
3.6.2.1	ifsent	9
3.6.2.2	pos	9
3.6.2.3	size	9
3.6.2.4	type	10
3.6.2.5	value	10
3.7	Smartphone Struct Reference	10
3.7.1	Constructor & Destructor Documentation	10
3.7.1.1	Smartphone(int A, float B, string(C), DatatbdeIM(D), DatasentM(E), Samples(F))	10

3.7.2	Member Data Documentation	10
3.7.2.1	battery	10
3.7.2.2	context	10
3.7.2.3	dtbd	11
3.7.2.4	dts	11
3.7.2.5	id_user	11
3.7.2.6	smp	11
3.8	User Struct Reference	11
3.8.1	Constructor & Destructor Documentation	11
3.8.1.1	User(int A, int B, Position C)	11
3.8.2	Member Data Documentation	11
3.8.2.1	id_pos	11
3.8.2.2	id_smartphone	12
3.8.2.3	pos	12
3.9	UserStat Struct Reference	12
3.9.1	Constructor & Destructor Documentation	12
3.9.1.1	UserStat(int A, int B, int C, double D)	12
3.9.2	Member Data Documentation	12
3.9.2.1	tot_data_gen	12
3.9.2.2	tot_en_tx	12
3.9.2.3	tot_energy	12
3.9.2.4	tot_num_samples	12

4 File Documentation	13
4.1 Headers/Antenna.h File Reference	13
4.1.1 Typedef Documentation	13
4.1.1.1 Antennas	13
4.1.2 Function Documentation	14
4.1.2.1 mapAntenna(string textFileAntennasLocations, string typeOfAntennas)	14
4.2 Headers/ClockManagement.h File Reference	14
4.2.1 Function Documentation	14
4.2.1.1 controlTime(tm ref_time_tm_, int minutes_trav_)	14
4.3 Headers/Event.h File Reference	14
4.3.1 Typedef Documentation	15
4.3.1.1 Events	15
4.3.1.2 GraphCityMap	15
4.3.2 Function Documentation	15
4.3.2.1 creatingListOfEvents(int num_users, Users usersM, Users::iterator it_usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCityMap::iterator it_grcM, string typeOfAntennas, int days)	15
4.3.2.2 creatingListOfEventsSlots(int num_users, Users usersM, Users::iterator it_usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCityMap::iterator it_grcM, string typeOfAntennas)	15
4.3.2.3 eventComparator(const Event &lhs, const Event &rhs)	15
4.3.2.4 readListOfEvents()	15
4.4 Headers/HeatMap.h File Reference	16
4.4.1 Function Documentation	16
4.4.1.1 heatMapAntennas(int num_users, Events eventsL, Antennas antennasL)	16
4.5 Headers/Location.h File Reference	16
4.5.1 Typedef Documentation	17
4.5.1.1 StreetPoint	17
4.6 Headers/MapCity.h File Reference	17
4.6.1 Typedef Documentation	17
4.6.1.1 GraphCityMap	17
4.6.2 Function Documentation	17

4.6.2.1	creatingMapAssociationGraphPoints(string mapPointsCentreCity, string nameCity)	17
4.6.2.2	dist(double th1, double ph1, double th2, double ph2)	18
4.7	Headers/Position.h File Reference	18
4.8	Headers/ReadValues.h File Reference	18
4.8.1	Function Documentation	18
4.8.1.1	allocateUsers(int num_users, StreetPoint stpM, StreetPoint::iterator it_stpM)	18
4.8.1.2	readDaySimulation()	19
4.8.1.3	readDecision()	19
4.8.1.4	readDefaultStreetPoint()	19
4.8.1.5	readFinishMinute()	19
4.8.1.6	readFinishtHour()	19
4.8.1.7	readKindAntennaFromSetupFile()	19
4.8.1.8	readMaximumTravelTime()	19
4.8.1.9	readMinimumTravelTime()	19
4.8.1.10	readNumberUsers()	19
4.8.1.11	readRay()	19
4.8.1.12	readStartHour()	19
4.8.1.13	readStartMinute()	19
4.8.1.14	readStreetPoint()	19
4.8.1.15	setDefaultMapCity()	19
4.8.1.16	setMapCity()	19
4.9	Headers/Sample.h File Reference	19
4.9.1	Typedef Documentation	20
4.9.1.1	Samples	20
4.10	Headers/Simulation.h File Reference	20
4.10.1	Function Documentation	20
4.10.1.1	simulationOperations(int num_users, Smartphones smM, Smartphones::iterator it_smM, Events eventsL, Events::iterator it_eventsL, Users usersM, Users::iterator it_usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCityMap::iterator it_grcM, int days)	20
4.11	Headers/Smartphones.h File Reference	20
4.11.1	Typedef Documentation	21

4.11.1.1	DatamentM	21
4.11.1.2	DatatbdeIM	21
4.11.1.3	Smartphones	21
4.11.2	Function Documentation	21
4.11.2.1	setSmartphones(int num_users)	21
4.12	Headers/Statistics.h File Reference	21
4.12.1	Function Documentation	21
4.12.1.1	computeStatistics(int num_users, Smartphones smM, Smartphones::iterator it↔ _smM, Events eventsL, Events::iterator it_eventsL, Users usersM, Users::iterator it_usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCityMap::iterator it_grcM, int days)	21
4.13	Headers/User.h File Reference	22
4.13.1	Typedef Documentation	22
4.13.1.1	Users	22
4.13.1.2	UsersStat	22
4.13.2	Function Documentation	22
4.13.2.1	allocateUsers(int num_users, StreetPoint stpM, StreetPoint::iterator it_stpM) . . .	22
4.13.2.2	allocateUsersSlotMobility(int num_users_tot, StreetPoint stpM, StreetPoint↔ ::iterator it_stpM)	23
4.14	Headers/Utilities.h File Reference	23
4.14.1	Macro Definition Documentation	24
4.14.1.1	LASTXN	24
4.14.1.2	MODULE	24
4.14.1.3	MYA	24
4.14.1.4	pi	24
4.14.1.5	R	24
4.14.1.6	RATIO	24
4.14.1.7	TO_RAD	24
4.14.1.8	UPTOMOD	24
4.14.2	Function Documentation	24
4.14.2.1	fRand(double fMin, double fMax)	24
4.14.2.2	havdist(double th1, double ph1, double th2, double ph2)	24

4.14.2.3	<code>PrintByteUnit(long double bytes)</code>	24
4.14.2.4	<code>PrintByteUnitInMiB(long double bytes)</code>	24
4.14.2.5	<code>rnd32(long seed)</code>	24
4.14.2.6	<code>uniform(double a, double b, long *seed)</code>	24
4.15	<code>src/Antenna.cc</code> File Reference	24
4.15.1	Function Documentation	25
4.15.1.1	<code>mapAntenna(string textFileAntennasLocations, string typeOfAntennas)</code>	25
4.15.2	Variable Documentation	25
4.15.2.1	<code>antennasLFunc</code>	25
4.15.2.2	<code>it_antennasLFunc</code>	25
4.16	<code>src/ClockManagement.cc</code> File Reference	25
4.16.1	Function Documentation	25
4.16.1.1	<code>controlTime(tm ref_time_, int minutes_trav_)</code>	25
4.17	<code>src/CrowdSenSim.cpp</code> File Reference	26
4.17.1	Function Documentation	26
4.17.1.1	<code>main(int argc, char **argv)</code>	26
4.17.2	Variable Documentation	27
4.17.2.1	<code>antennasL</code>	27
4.17.2.2	<code>days</code>	27
4.17.2.3	<code>eventsL</code>	27
4.17.2.4	<code>grcM</code>	27
4.17.2.5	<code>it_antennasL</code>	27
4.17.2.6	<code>it_eventsL</code>	27
4.17.2.7	<code>it_grcM</code>	27
4.17.2.8	<code>it_smM</code>	27
4.17.2.9	<code>it_stpM</code>	27
4.17.2.10	<code>it_usersM</code>	27
4.17.2.11	<code>it_usersstatM</code>	27
4.17.2.12	<code>num_users</code>	27
4.17.2.13	<code>smM</code>	27

4.17.2.14 stpM	27
4.17.2.15 usersM	27
4.17.2.16 usersstatM	27
4.18 src/Event.cc File Reference	27
4.18.1 Function Documentation	28
4.18.1.1 creatingListOfEvents(int num_users, Users usersM, Users::iterator it_usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCity↔ Map::iterator it_grcM, string typeOfAntennas, int days)	28
4.18.1.2 creatingListOfEventsSlots(int num_users, Users usersM, Users::iterator it_↔ usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCityMap::iterator it_grcM, string typeOfAntennas)	28
4.18.1.3 eventComparator(const Event &lhs, const Event &rhs)	28
4.18.1.4 readListOfEvents()	28
4.18.2 Variable Documentation	28
4.18.2.1 eventsLFunc	28
4.18.2.2 it_eventsLFunc	28
4.19 src/HeatMap.cc File Reference	28
4.19.1 Function Documentation	29
4.19.1.1 heatMapAntennas(int num_users, Events eventsL, Antennas antennasL)	29
4.20 src/Location.cc File Reference	29
4.21 src/MapCity.cc File Reference	29
4.21.1 Macro Definition Documentation	30
4.21.1.1 pi	30
4.21.1.2 R	30
4.21.1.3 TO_RAD	30
4.21.2 Function Documentation	30
4.21.2.1 creatingMapAssociationGraphPoints(string mapPointsCentreCity, string nameCity)	30
4.21.2.2 dist(double th1, double ph1, double th2, double ph2)	30
4.21.3 Variable Documentation	31
4.21.3.1 it_locationsLA	31
4.21.3.2 locationsLA	31
4.22 src/Position.cc File Reference	31

4.23	src/ReadValues.cc File Reference	31
4.23.1	Function Documentation	31
4.23.1.1	readDaySimulation()	31
4.23.1.2	readDecision()	31
4.23.1.3	readDefaultStreetPoint()	31
4.23.1.4	readFinishMinute()	31
4.23.1.5	readFinishtHour()	32
4.23.1.6	readKindAntennaFromSetupFile()	32
4.23.1.7	readMaximumTravelTime()	32
4.23.1.8	readMinimumTravelTime()	32
4.23.1.9	readNumberUsers()	32
4.23.1.10	readRay()	32
4.23.1.11	readStartHour()	32
4.23.1.12	readStartMinute()	32
4.23.1.13	readStreetPoint()	32
4.23.1.14	setDefaultMapCity()	32
4.23.1.15	setMapCity()	32
4.24	src/Sample.cc File Reference	32
4.25	src/Simulation.cc File Reference	32
4.25.1	Function Documentation	33
4.25.1.1	simulationOperations(int num_users, Smartphones smM, Smartphones::iterator it_smM, Events eventsL, Events::iterator it_eventsL, Users usersM, Users::iterator it_usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCityMap::iterator it_grcM, int days)	33
4.26	src/Smartphones.cc File Reference	33
4.26.1	Function Documentation	33
4.26.1.1	setSmartphones(int num_users)	33
4.27	src/Statistics.cc File Reference	33
4.27.1	Function Documentation	33
4.27.1.1	computeStatistics(int num_users, Smartphones smM, Smartphones::iterator it_smM, Events eventsL, Events::iterator it_eventsL, Users usersM, Users::iterator it_usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCityMap::iterator it_grcM, int days)	33

4.28	src/User.cc File Reference	34
4.28.1	Function Documentation	34
4.28.1.1	allocateUsers(int num_users, StreetPoint stpM, StreetPoint::iterator it_stpM)	34
4.28.1.2	allocateUsersSlotMobility(int num_users_tot, StreetPoint stpM, StreetPoint::iterator it_stpM)	34
4.29	src/Utilities.cc File Reference	34
4.29.1	Macro Definition Documentation	35
4.29.1.1	pi	35
4.29.1.2	R	35
4.29.1.3	TO_RAD	35
4.29.2	Function Documentation	35
4.29.2.1	fRand(double fMin, double fMax)	35
4.29.2.2	havdist(double th1, double ph1, double th2, double ph2)	35
4.29.2.3	PrintByteUnit(long double bytes)	35
4.29.2.4	PrintByteUnitInMiB(long double bytes)	35
4.29.2.5	rnd32(long seed)	35
4.29.2.6	uniform(double a, double b, long *seed)	35
	Index	37

Chapter 1

Class Index

1.1 Class List

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

Antenna	5
Event	6
Location	7
LocationA	7
Position	8
Sample	9
Smartphone	10
User	11
UserStat	12

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Headers/ Antenna.h	13
Headers/ ClockManagement.h	14
Headers/ Event.h	14
Headers/ HeatMap.h	16
Headers/ Location.h	16
Headers/ MapCity.h	17
Headers/ Position.h	18
Headers/ ReadValues.h	18
Headers/ Sample.h	19
Headers/ Simulation.h	20
Headers/ Smartphones.h	20
Headers/ Statistics.h	21
Headers/ User.h	22
Headers/ Utilities.h	23
src/ Antenna.cc	24
src/ ClockManagement.cc	25
src/ CrowdSenSim.cpp	26
src/ Event.cc	27
src/ HeatMap.cc	28
src/ Location.cc	29
src/ MapCity.cc	29
src/ Position.cc	31
src/ ReadValues.cc	31
src/ Sample.cc	32
src/ Simulation.cc	32
src/ Smartphones.cc	33
src/ Statistics.cc	33
src/ User.cc	34
src/ Utilities.cc	34

Chapter 3

Class Documentation

3.1 Antenna Struct Reference

```
#include <Antenna.h>
```

Public Member Functions

- [Antenna](#) (int *A*, string *B*, [Location](#) *C*)

Public Attributes

- int [id_antenna](#)
- string [type_antenna](#)
- [Location](#) *loc*

3.1.1 Constructor & Destructor Documentation

3.1.1.1 [Antenna::Antenna](#) (int *A*, string *B*, [Location](#) *C*) `[inline]`

3.1.2 Member Data Documentation

3.1.2.1 int [Antenna::id_antenna](#)

[Antenna](#) ID

3.1.2.2 [Location](#) [Antenna::loc](#)

Geographical position of the [Antenna](#)

3.1.2.3 string Antenna::type_antenna

Type of antenna

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

- Headers/[Antenna.h](#)

3.2 Event Struct Reference

```
#include <Event.h>
```

Public Member Functions

- [Event](#) (int A, [Location](#) B, tm C)

Public Attributes

- int [id_user](#)
- [Location](#) [loc](#)
- tm [timestamp](#)

3.2.1 Constructor & Destructor Documentation

3.2.1.1 [Event::Event](#) (int A, [Location](#) B, tm C) `[inline]`

3.2.2 Member Data Documentation

3.2.2.1 int [Event::id_user](#)

[User](#) ID

3.2.2.2 [Location](#) [Event::loc](#)

[Location](#) of the event

3.2.2.3 tm [Event::timestamp](#)

Time of the event

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

- Headers/[Event.h](#)

3.3 Location Struct Reference

```
#include <Location.h>
```

Public Member Functions

- [Location](#) (float A, float B, float C)

Public Attributes

- float [lat](#)
- float [lon](#)
- float [alt](#)

3.3.1 Constructor & Destructor Documentation

3.3.1.1 [Location::Location](#) (float A, float B, float C) `[inline]`

3.3.2 Member Data Documentation

3.3.2.1 float [Location::alt](#)

Altitude - Geo

3.3.2.2 float [Location::lat](#)

Latitude- Geo

3.3.2.3 float [Location::lon](#)

Longitude - Geo

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

- Headers/[Location.h](#)

3.4 LocationA Struct Reference

Public Member Functions

- [LocationA](#) (int A, double B, double C, double D)

Public Attributes

- int [id](#)
- double [lat](#)
- double [lon](#)
- double [alt](#)

3.4.1 Constructor & Destructor Documentation

3.4.1.1 `LocationA::LocationA (int A, double B, double C, double D)` `[inline]`

3.4.2 Member Data Documentation

3.4.2.1 `double LocationA::alt`

3.4.2.2 `int LocationA::id`

3.4.2.3 `double LocationA::lat`

3.4.2.4 `double LocationA::lon`

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

- `src/MapCity.cc`

3.5 Position Struct Reference

```
#include <Position.h>
```

Public Member Functions

- [Position](#) ([Location](#) *A*, tm *B*)

Public Attributes

- [Location](#) *loc*
- tm [timestamp](#)

3.5.1 Constructor & Destructor Documentation

3.5.1.1 `Position::Position (Location A, tm B)` `[inline]`

3.5.2 Member Data Documentation

3.5.2.1 `Location Position::loc`

Geographic location

3.5.2.2 tm Position::timestamp

Time

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

- Headers/[Position.h](#)

3.6 Sample Struct Reference

```
#include <Sample.h>
```

Public Member Functions

- [Sample](#) (string *A*, float *B*, float *C*, bool *D*, [Position](#) *E*)

Public Attributes

- string [type](#)
- float [value](#)
- float [size](#)
- bool [ifsent](#)
- [Position](#) [pos](#)

3.6.1 Constructor & Destructor Documentation

3.6.1.1 `Sample::Sample (string A, float B, float C, bool D, Position E)` `[inline]`

3.6.2 Member Data Documentation

3.6.2.1 `bool Sample::ifsent`

Sent or no

3.6.2.2 `Position Sample::pos`

Geo-temporal position

3.6.2.3 `float Sample::size`

Size of data to be delivered (Bytes)

3.6.2.4 string Sample::type

Type of the sensor

3.6.2.5 float Sample::value

Value of data

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

- Headers/[Sample.h](#)

3.7 Smartphone Struct Reference

```
#include <Smartphones.h>
```

Public Member Functions

- [Smartphone](#) (int A, float B, string(C), [DatatbdeIM](#)(D), [DatasantM](#)(E), [Samples](#)(F))

Public Attributes

- int [id_user](#)
- float [battery](#)
- string [context](#)
- [DatatbdeIM](#) dtbd
- [DatasantM](#) dts
- [Samples](#) smp

3.7.1 Constructor & Destructor Documentation

3.7.1.1 [Smartphone::Smartphone](#) (int A, float B, string(C) , [DatatbdeIM](#)(D) , [DatasantM](#)(E) , [Samples](#)(F))
[inline]

3.7.2 Member Data Documentation

3.7.2.1 float Smartphone::battery

Battery Level (0-100)

3.7.2.2 string Smartphone::context

Context (outdoor, semi-outdoor, indoor)

3.7.2.3 Data**tdel**M Smartphone::dtbd

Amount of data to be delivered

3.7.2.4 Data**sent**M Smartphone::dts

Amount of data already sent

3.7.2.5 int Smartphone::id_user

User ID

3.7.2.6 Sample**s** Smartphone::smp

List of samples

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

- Headers/[Smartphones.h](#)

3.8 User Struct Reference

```
#include <User.h>
```

Public Member Functions

- [User](#) (int A, int B, [Position](#) C)

Public Attributes

- int [id_smartphone](#)
- int [id_pos](#)
- [Position](#) pos

3.8.1 Constructor & Destructor Documentation

3.8.1.1 User::User (int A, int B, [Position](#) C) `[inline]`

3.8.2 Member Data Documentation

3.8.2.1 int User::id_pos

ID of the position

3.8.2.2 int User::id_smartphone

Smartphone ID

3.8.2.3 Position User::pos

Geo-temporal position

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

- Headers/[User.h](#)

3.9 UserStat Struct Reference

```
#include <User.h>
```

Public Member Functions

- [UserStat](#) (int A, int B, int C, double D)

Public Attributes

- int [tot_num_samples](#)
- int [tot_data_gen](#)
- int [tot_energy](#)
- double [tot_en_tx](#)

3.9.1 Constructor & Destructor Documentation

3.9.1.1 [UserStat::UserStat](#) (int A, int B, int C, double D) [[inline](#)]

3.9.2 Member Data Documentation

3.9.2.1 int UserStat::tot_data_gen

Amount of data generated

3.9.2.2 double UserStat::tot_en_tx

Total energy spent for transmission

3.9.2.3 int UserStat::tot_energy

Total energy spent

3.9.2.4 int UserStat::tot_num_samples

of samples generated

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

- Headers/[User.h](#)

Chapter 4

File Documentation

4.1 Headers/Antenna.h File Reference

```
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
#include "Location.h"
```

Classes

- struct [Antenna](#)

Typedefs

- typedef std::list< [Antenna](#) > [Antennas](#)

Functions

- [Antennas mapAntenna](#) (string textFileAntennasLocations, string typeOfAntennas)

4.1.1 Typedef Documentation

4.1.1.1 typedef std::list<Antenna> Antennas

List of antenna

4.1.2 Function Documentation

4.1.2.1 Antennas mapAntenna (string *textFileAntennasLocations*, string *typeOfAntennas*)

Creates antennas-map.txt file

Inputs: -file of geographical location of Antennas -Name of the kind of antennas system

Returns a list of [Antenna](#) It sets antennas in the city map

4.2 Headers/ClockManagement.h File Reference

```
#include <time.h>
```

Functions

- tm [controlTime](#) (tm *ref_time_tm_*, int *minutes_trav_*)

4.2.1 Function Documentation

4.2.1.1 tm controlTime (tm *ref_time_tm_*, int *minutes_trav_*)

Updates the current value of *ref_time_tm*

Inputs: -*ref_time_tm*: time to update -*minutes_trav_*: minutes to add

Returns an object of the struct tm It checks the travel time

4.3 Headers/Event.h File Reference

```
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
#include <time.h>
#include "../Headers/Utilities.h"
#include "../Headers/ClockManagement.h"
```

Classes

- struct [Event](#)

Typedefs

- typedef std::list< [Event](#) > [Events](#)
- typedef std::map< int, list< int > > [GraphCityMap](#)

Functions

- bool [eventComparator](#) (const [Event](#) &lhs, const [Event](#) &rhs)
- [Events](#) [creatingListOfEvents](#) (int num_users, [Users](#) usersM, [Users::iterator](#) it_usersM, [StreetPoint](#) stpM, [StreetPoint::iterator](#) it_stpM, [GraphCityMap](#) grcM, [GraphCityMap::iterator](#) it_grcM, string typeOfAntennas, int days)
- [Events](#) [creatingListOfEventsSlots](#) (int num_users, [Users](#) usersM, [Users::iterator](#) it_usersM, [StreetPoint](#) stpM, [StreetPoint::iterator](#) it_stpM, [GraphCityMap](#) grcM, [GraphCityMap::iterator](#) it_grcM, string typeOfAntennas)
- [Events](#) [readListOfEvents](#) ()

4.3.1 Typedef Documentation

4.3.1.1 typedef std::list<Event> Events

4.3.1.2 typedef std::map<int, list<int> > GraphCityMap

4.3.2 Function Documentation

4.3.2.1 Events [creatingListOfEvents](#) (int num_users, [Users](#) usersM, [Users::iterator](#) it_usersM, [StreetPoint](#) stpM, [StreetPoint::iterator](#) it_stpM, [GraphCityMap](#) grcM, [GraphCityMap::iterator](#) it_grcM, string typeOfAntennas, int days)

It creates lists of events, ordered by UserID and time It creates a list of events evenly in the time. It generates the file of events

4.3.2.2 Events [creatingListOfEventsSlots](#) (int num_users, [Users](#) usersM, [Users::iterator](#) it_usersM, [StreetPoint](#) stpM, [StreetPoint::iterator](#) it_stpM, [GraphCityMap](#) grcM, [GraphCityMap::iterator](#) it_grcM, string typeOfAntennas)

It creates lists of events, ordered by UserID and time slots

4.3.2.3 bool [eventComparator](#) (const [Event](#) & lhs, const [Event](#) & rhs)

It compares two Events Inputs: Events lhs and Events rhs Output: boolean value of the comparison It makes a time comparison between events, and return a boolean value

4.3.2.4 Events [readListOfEvents](#) ()

It reads lists of events

4.4 Headers/HeatMap.h File Reference

```
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
#include <time.h>
#include "../Headers/Utilities.h"
```

Functions

- void [heatMapAntennas](#) (int [num_users](#), [Events eventsL](#), [Antennas antennasL](#))

4.4.1 Function Documentation

4.4.1.1 void [heatMapAntennas](#) (int *num_users*, *Events eventsL*, *Antennas antennasL*)

It creates the "HeatMapPosition.txt" file, useful for the Heat Map on the statistics webpage

Inputs: -num_users: total number of users -eventsL: list of simulation events -antennasL: list of antennas in the city

4.5 Headers/Location.h File Reference

```
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
```

Classes

- struct [Location](#)

Typedefs

- typedef map< int, [Location](#) > [StreetPoint](#)

4.5.1 Typedef Documentation

4.5.1.1 typedef map<int,Location> StreetPoint

4.6 Headers/MapCity.h File Reference

```
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
```

Typedefs

- typedef std::map< int, list< int > > [GraphCityMap](#)

Functions

- double [dist](#) (double th1, double ph1, double th2, double ph2)
- void [creatingMapAssociationGraphPoints](#) (string mapPointsCentreCity, string nameCity)

4.6.1 Typedef Documentation

4.6.1.1 typedef std::map<int, list<int> > GraphCityMap

4.6.2 Function Documentation

4.6.2.1 void creatingMapAssociationGraphPoints (string *mapPointsCentreCity*, string *nameCity*)

It creates two different files:

- MapGraphPoints.txt: it contains a list of points and every adjacent points, where users can move. The points including within the ray are called adjacent
- MapAssociation.txt: it contains the list of all points (with geographical references) where the pedestrians move.

4.6.2.2 double dist (double *th1*, double *ph1*, double *th2*, double *ph2*)

It computes the distance between two general points in the map Inputs: Longitude and latitude of two points
Outputs: double value of the distance Computes the distance in meters between two points in the map.

4.7 Headers/Position.h File Reference

```
#include <time.h>
#include "Location.h"
```

Classes

- struct [Position](#)

4.8 Headers/ReadValues.h File Reference

```
#include "../Headers/Utilities.h"
```

Functions

- [StreetPoint](#) readStreetPoint ()
- [GraphCityMap](#) setMapCity ()
- [StreetPoint](#) readDefaultStreetPoint ()
- [GraphCityMap](#) setDefaultMapCity ()
- [Users](#) allocateUsers (int num_users, [StreetPoint](#) stpM, [StreetPoint::iterator](#) it_stpM)
- int readDaySimulation ()
- int readNumberUsers ()
- string readKindAntennaFromSetupFile ()
- int readDecision ()
- int readRay ()
- int readStartHour ()
- int readStartMinute ()
- int readFinishtHour ()
- int readFinishMinute ()
- int readMaximumTravelTime ()
- int readMinimumTravelTime ()

4.8.1 Function Documentation

4.8.1.1 Users allocateUsers (int num_users, [StreetPoint](#) stpM, [StreetPoint::iterator](#) it_stpM)

It allocates users in the map. The first position is chosen casually.

4.8.1.2 int readDaySimulation ()

4.8.1.3 int readDecision ()

4.8.1.4 **StreetPoint** readDefaultStreetPoint ()

4.8.1.5 int readFinishMinute ()

4.8.1.6 int readFinishtHour ()

4.8.1.7 string readKindAntennaFromSetupFile ()

4.8.1.8 int readMaximumTravelTime ()

4.8.1.9 int readMinimumTravelTime ()

4.8.1.10 int readNumberUsers ()

4.8.1.11 int readRay ()

4.8.1.12 int readStartHour ()

4.8.1.13 int readStartMinute ()

4.8.1.14 **StreetPoint** readStreetPoint ()

These functions read values useful to starting the Simulation from the Setup.txt file, located in the Input folder

4.8.1.15 **GraphCityMap** setDefaultMapCity ()

4.8.1.16 **GraphCityMap** setMapCity ()

4.9 Headers/Sample.h File Reference

```
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
#include "Position.h"
```

Classes

- struct [Sample](#)

Typedefs

- typedef std::list< [Sample](#) > [Samples](#)

4.9.1 Typedef Documentation

4.9.1.1 typedef std::list<Sample> Samples

4.10 Headers/Simulation.h File Reference

```
#include "../Headers/Utilities.h"
```

Functions

- void [simulationOperations](#) (int [num_users](#), [Smartphones](#) [smM](#), [Smartphones::iterator](#) [it_smM](#), [Events](#) [eventsL](#), [Events::iterator](#) [it_eventsL](#), [Users](#) [usersM](#), [Users::iterator](#) [it_usersM](#), [StreetPoint](#) [stpM](#), [StreetPoint::iterator](#) [it_stpM](#), [GraphCityMap](#) [grcM](#), [GraphCityMap::iterator](#) [it_grcM](#), int [days](#))

4.10.1 Function Documentation

4.10.1.1 void simulationOperations (int num_users, Smartphones smM, Smartphones::iterator it_smM, Events eventsL, Events::iterator it_eventsL, Users usersM, Users::iterator it_usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCityMap::iterator it_grcM, int days)

It creates the SimulationData.txt file useful to compute the statistics and it is the core of the simulator.

4.11 Headers/Smartphones.h File Reference

```
#include "Sample.h"
```

Classes

- struct [Smartphone](#)

Typedefs

- typedef std::map< string, float > [DatatbdeIM](#)
- typedef std::map< string, float > [DatasantM](#)
- typedef std::map< int, [Smartphone](#) > [Smartphones](#)

Functions

- [Smartphones setSmartphones](#) (int num_users)

4.11.1 Typedef Documentation

4.11.1.1 typedef std::map<string, float> DatasentM

The amount of data already sent

4.11.1.2 typedef std::map<string, float> DatatbdeIM

The amount of data to be delivered

4.11.1.3 typedef std::map<int, Smartphone> Smartphones

4.11.2 Function Documentation

4.11.2.1 Smartphones setSmartphones (int num_users)

It sets the details of the smartphone for each user

4.12 Headers/Statistics.h File Reference

```
#include "../Headers/Utilities.h"
```

Functions

- void [computeStatistics](#) (int num_users, [Smartphones smM](#), [Smartphones::iterator it_smM](#), [Events eventsL](#), [Events::iterator it_eventsL](#), [Users usersM](#), [Users::iterator it_usersM](#), [StreetPoint stpM](#), [StreetPoint::iterator it_stpM](#), [GraphCityMap grcM](#), [GraphCityMap::iterator it_grcM](#), int days)

4.12.1 Function Documentation

4.12.1.1 void computeStatistics (int num_users, [Smartphones smM](#), [Smartphones::iterator it_smM](#), [Events eventsL](#), [Events::iterator it_eventsL](#), [Users usersM](#), [Users::iterator it_usersM](#), [StreetPoint stpM](#), [StreetPoint::iterator it_stpM](#), [GraphCityMap grcM](#), [GraphCityMap::iterator it_grcM](#), int days)

It computes the statistics parameters from the SimulationData.txt file Delimiter constants for reading file.

4.13 Headers/User.h File Reference

```
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
#include "Position.h"
```

Classes

- struct [User](#)
- struct [UserStat](#)

Typedefs

- typedef std::map< int, [User](#) > [Users](#)
- typedef std::map< int, [UserStat](#) > [UsersStat](#)

Functions

- [Users allocateUsers](#) (int [num_users](#), [StreetPoint](#) [stpM](#), [StreetPoint::iterator](#) [it_stpM](#))
- [Users allocateUsersSlotMobility](#) (int [num_users_tot](#), [StreetPoint](#) [stpM](#), [StreetPoint::iterator](#) [it_stpM](#))

4.13.1 Typedef Documentation

4.13.1.1 typedef std::map<int,[User](#)> [Users](#)

4.13.1.2 typedef std::map<int,[UserStat](#)> [UsersStat](#)

4.13.2 Function Documentation

4.13.2.1 [Users allocateUsers](#) (int [num_users](#), [StreetPoint](#) [stpM](#), [StreetPoint::iterator](#) [it_stpM](#))

It allocates the users in the map. More in details the user are allocated evenly in the timestamp set from [User](#) It allocates users in the map. The first position is chosen casually.

4.13.2.2 Users allocateUsersSlotMobility (int num_users_tot, StreetPoint stpM, StreetPoint::iterator it_stpM)

It allocates the users in the map. More in details the number of users allocated depends on the probability of presence for every time slot. It allocates users in different slots time, according to the presence probability. If it is higher there are many users in that slot. There are 10 slots and each one lasts 1 hour. If it chosen this kind of users allocation, the simulation starts at 21 and finish at 7.

4.14 Headers/Utilities.h File Reference

```
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
#include "../Headers/Position.h"
#include "../Headers/Location.h"
#include "../Headers/Sample.h"
#include "../Headers/Smartphones.h"
#include "../Headers/User.h"
#include "../Headers/Antenna.h"
#include "../Headers/Event.h"
#include "../Headers/MapCity.h"
```

Macros

- #define MODULE 2147483647
- #define MYA 16807
- #define LASTXN 127773
- #define UPTOMOD -2836
- #define RATIO 0.46566128e-9
- #define pi 3.14159265358979323846
- #define R 6371
- #define TO_RAD (3.1415926536 / 180)

Functions

- long rnd32 (long seed)
- double uniform (double a, double b, long *seed)
- double fRand (double fMin, double fMax)
- double havdist (double th1, double ph1, double th2, double ph2)
- std::string PrintByteUnit (long double bytes)
- std::string PrintByteUnitInMiB (long double bytes)

4.14.1 Macro Definition Documentation

4.14.1.1 `#define LASTXN 127773`

4.14.1.2 `#define MODULE 2147483647`

4.14.1.3 `#define MYA 16807`

4.14.1.4 `#define pi 3.14159265358979323846`

4.14.1.5 `#define R 6371`

4.14.1.6 `#define RATIO 0.46566128e-9`

4.14.1.7 `#define TO_RAD (3.1415926536 / 180)`

4.14.1.8 `#define UPTOMOD -2836`

4.14.2 Function Documentation

4.14.2.1 `double fRand (double fMin, double fMax)`

4.14.2.2 `double havdist (double th1, double ph1, double th2, double ph2)`

It computes the distance in meters between two different points, given the Latitude and Longitude coordinates.

4.14.2.3 `std::string PrintByteUnit (long double bytes)`

It converts bytes values in KB/MB/GB automatically.

4.14.2.4 `std::string PrintByteUnitInMiB (long double bytes)`

It converts byte values in MiB.

4.14.2.5 `long rnd32 (long seed)`

4.14.2.6 `double uniform (double a, double b, long * seed)`

4.15 src/Antenna.cc File Reference

```
#include "../Headers/Antenna.h"
#include <string>
#include <fstream>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
```

Functions

- [Antennas mapAntenna](#) (string *textFileAntennasLocations*, string *typeOfAntennas*)

Variables

- [Antennas antennasLFunc](#)
- [Antennas::iterator it_antennasLFunc](#)

4.15.1 Function Documentation

4.15.1.1 [Antennas mapAntenna](#) (string *textFileAntennasLocations*, string *typeOfAntennas*)

Creates antennas-map.txt file

Inputs: -file of geographical location of Antennas -Name of the kind of antennas system

Returns a list of [Antenna](#) It sets antennas in the city map

4.15.2 Variable Documentation

4.15.2.1 [Antennas antennasLFunc](#)

4.15.2.2 [Antennas::iterator it_antennasLFunc](#)

4.16 src/ClockManagement.cc File Reference

```
#include "../Headers/ClockManagement.h"
#include <time.h>
```

Functions

- tm [controlTime](#) (tm *ref_time_*, int *minutes_trav_*)

4.16.1 Function Documentation

4.16.1.1 [tm controlTime](#) (tm *ref_time_tm_*, int *minutes_trav_*)

Updates the current value of *ref_time_tm*

Inputs: -*ref_time_tm*: time to update -*minutes_trav_*: minutes to add

Returns an object of the struct tm It checks the travel time

4.17 src/CrowdSenSim.cpp File Reference

```
#include <iostream>
#include "../Headers/Position.h"
#include "../Headers/Location.h"
#include "../Headers/Sample.h"
#include "../Headers/Antenna.h"
#include "../Headers/Event.h"
#include "../Headers/MapCity.h"
#include "../Headers/ReadValues.h"
#include "../Headers/Simulation.h"
#include "../Headers/Smartphones.h"
#include "../Headers/Statistics.h"
#include "../Headers/User.h"
#include "../Headers/Utilities.h"
#include "../Headers/HeatMap.h"
```

Functions

- `int main (int argc, char **argv)`

Variables

- `Smartphones smM`
- `Smartphones::iterator it_smM`
- `Users usersM`
- `Users::iterator it_usersM`
- `UsersStat usersstatM`
- `UsersStat::iterator it_usersstatM`
- `Antennas antennasL`
- `Antennas::iterator it_antennasL`
- `StreetPoint stpM`
- `StreetPoint::iterator it_stpM`
- `Events eventsL`
- `Events::iterator it_eventsL`
- `GraphCityMap grcM`
- `GraphCityMap::iterator it_grcM`
- `int days`
- `int num_users`

4.17.1 Function Documentation

4.17.1.1 `int main (int argc, char ** argv)`

Decision for creating or not a new list of events.

Which kind of mobility is using.

This is the case it is not using a slot mobility

4.17.2 Variable Documentation

4.17.2.1 **Antennas** `antennasL`

4.17.2.2 `int` `days`

4.17.2.3 **Events** `eventsL`

4.17.2.4 **GraphCityMap** `grcM`

4.17.2.5 `Antennas::iterator` `it_antennasL`

4.17.2.6 `Events::iterator` `it_eventsL`

4.17.2.7 `GraphCityMap::iterator` `it_grcM`

4.17.2.8 `Smartphones::iterator` `it_smM`

4.17.2.9 `StreetPoint::iterator` `it_stpM`

4.17.2.10 `Users::iterator` `it_usersM`

4.17.2.11 `UsersStat::iterator` `it_usersstatM`

4.17.2.12 `int` `num_users`

4.17.2.13 **Smartphones** `smM`

4.17.2.14 **StreetPoint** `stpM`

4.17.2.15 **Users** `usersM`

4.17.2.16 **UsersStat** `usersstatM`

4.18 src/Event.cc File Reference

```
#include "../Headers/Event.h"
#include <iostream>
#include <string>
#include <fstream>
#include "../Headers/Antenna.h"
#include "../Headers/ClockManagement.h"
#include "../Headers/ReadValues.h"
```

Functions

- bool `eventComparator` (const `Event` &lhs, const `Event` &rhs)
- `Events` `creatingListOfEvents` (int `num_users`, `Users` `usersM`, `Users::iterator` `it_usersM`, `StreetPoint` `stpM`, `StreetPoint::iterator` `it_stpM`, `GraphCityMap` `grcM`, `GraphCityMap::iterator` `it_grcM`, string `typeOfAntennas`, int `days`)
- `Events` `readListOfEvents` ()
- `Events` `creatingListOfEventsSlots` (int `num_users`, `Users` `usersM`, `Users::iterator` `it_usersM`, `StreetPoint` `stpM`, `StreetPoint::iterator` `it_stpM`, `GraphCityMap` `grcM`, `GraphCityMap::iterator` `it_grcM`, string `typeOfAntennas`)

Variables

- `Events` `eventsLFunc`
- `Events::iterator` `it_eventsLFunc`

4.18.1 Function Documentation

4.18.1.1 `Events` `creatingListOfEvents` (int `num_users`, `Users` `usersM`, `Users::iterator` `it_usersM`, `StreetPoint` `stpM`, `StreetPoint::iterator` `it_stpM`, `GraphCityMap` `grcM`, `GraphCityMap::iterator` `it_grcM`, string `typeOfAntennas`, int `days`)

It creates lists of events, ordered by UserID and time It creates a list of events evenly in the time. It generates the file of events

4.18.1.2 `Events` `creatingListOfEventsSlots` (int `num_users`, `Users` `usersM`, `Users::iterator` `it_usersM`, `StreetPoint` `stpM`, `StreetPoint::iterator` `it_stpM`, `GraphCityMap` `grcM`, `GraphCityMap::iterator` `it_grcM`, string `typeOfAntennas`)

It creates lists of events, ordered by UserID and time slots

4.18.1.3 `bool` `eventComparator` (const `Event` & `lhs`, const `Event` & `rhs`)

It compares two Events Inputs: Events `lhs` and Events `rhs` Output: boolean value of the comparison It makes a time comparison between events, and return a boolean value

4.18.1.4 `Events` `readListOfEvents` ()

It reads lists of events

4.18.2 Variable Documentation

4.18.2.1 `Events` `eventsLFunc`

4.18.2.2 `Events::iterator` `it_eventsLFunc`

4.19 `src/HeatMap.cc` File Reference

```
#include "../Headers/HeatMap.h"
```


Functions

- void [heatMapAntennas](#) (int [num_users](#), [Events eventsL](#), [Antennas antennasL](#))

4.19.1 Function Documentation

4.19.1.1 void [heatMapAntennas](#) (int *num_users*, *Events eventsL*, *Antennas antennasL*)

It creates the "HeatMapPosition.txt" file, useful for the Heat Map on the statistics webpage

Inputs: -num_users: total number of users -eventsL: list of simulation events -antennasL: list of antennas in the city

4.20 src/Location.cc File Reference

```
#include "../Headers/Location.h"
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
```

4.21 src/MapCity.cc File Reference

```
#include "../Headers/MapCity.h"
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
#include "../Headers/Event.h"
#include "../Headers/ReadValues.h"
```

Classes

- struct [LocationA](#)

Macros

- `#define pi 3.14159265358979323846`
- `#define R 6371`
- `#define TO_RAD (3.1415926536 / 180)`

Functions

- `double dist (double th1, double ph1, double th2, double ph2)`
- `void creatingMapAssociationGraphPoints (string mapPointsCentreCity, string nameCity)`

Variables

- `list< LocationA > locationsLA`
- `list< LocationA >::iterator it_locationsLA`

4.21.1 Macro Definition Documentation

4.21.1.1 `#define pi 3.14159265358979323846`

4.21.1.2 `#define R 6371`

4.21.1.3 `#define TO_RAD (3.1415926536 / 180)`

4.21.2 Function Documentation

4.21.2.1 `void creatingMapAssociationGraphPoints (string mapPointsCentreCity, string nameCity)`

It creates two different files:

- `MapGraphPoints.txt`: it contains a list of points and every adjacent points, where users can move. The points including within the ray are called adjacent
- `MapAssociation.txt`: it contains the list of all points (with geographical references) where the pedestrians move.

4.21.2.2 `double dist (double th1, double ph1, double th2, double ph2)`

It computes the distance between two general points in the map
 Inputs: Longitude and latitude of two points
 Outputs: double value of the distance
 Computes the distance in meters between two points in the map.

4.21.3 Variable Documentation

4.21.3.1 `list<LocationA>::iterator it_locationsLA`

4.21.3.2 `list<LocationA> locationsLA`

4.22 src/Position.cc File Reference

```
#include "../Headers/Position.h"
```

4.23 src/ReadValues.cc File Reference

```
#include "../Headers/ReadValues.h"  
#include <string.h>  
#include <sstream>
```

Functions

- [StreetPoint readStreetPoint \(\)](#)
- [GraphCityMap setMapCity \(\)](#)
- [StreetPoint readDefaultStreetPoint \(\)](#)
- [GraphCityMap setDefaultMapCity \(\)](#)
- [int readDaySimulation \(\)](#)
- [int readNumberUsers \(\)](#)
- [string readKindAntennaFromSetupFile \(\)](#)
- [int readDecision \(\)](#)
- [int readRay \(\)](#)
- [int readStartHour \(\)](#)
- [int readStartMinute \(\)](#)
- [int readFinishtHour \(\)](#)
- [int readFinishMinute \(\)](#)
- [int readMinimumTravelTime \(\)](#)
- [int readMaximumTravelTime \(\)](#)

4.23.1 Function Documentation

4.23.1.1 `int readDaySimulation ()`

4.23.1.2 `int readDecision ()`

4.23.1.3 `StreetPoint readDefaultStreetPoint ()`

4.23.1.4 `int readFinishMinute ()`

4.23.1.5 `int readFinishtHour ()`

4.23.1.6 `string readKindAntennaFromSetupFile ()`

4.23.1.7 `int readMaximumTravelTime ()`

4.23.1.8 `int readMinimumTravelTime ()`

4.23.1.9 `int readNumberUsers ()`

4.23.1.10 `int readRay ()`

4.23.1.11 `int readStartHour ()`

4.23.1.12 `int readStartMinute ()`

4.23.1.13 `StreetPoint readStreetPoint ()`

These functions read values useful to starting the Simulation from the Setup.txt file, located in the Input folder

4.23.1.14 `GraphCityMap setDefaultMapCity ()`

4.23.1.15 `GraphCityMap setMapCity ()`

4.24 `src/Sample.cc` File Reference

```
#include "../Headers/Sample.h"
```

4.25 `src/Simulation.cc` File Reference

```
#include "../Headers/Simulation.h"
#include "../Headers/Utilities.h"
```

Functions

- void [simulationOperations](#) (int [num_users](#), [Smartphones smM](#), [Smartphones::iterator it_smM](#), [Events eventsL](#), [Events::iterator it_eventsL](#), [Users usersM](#), [Users::iterator it_usersM](#), [StreetPoint stpM](#), [StreetPoint::iterator it_stpM](#), [GraphCityMap grcM](#), [GraphCityMap::iterator it_grcM](#), int [days](#))

4.25.1 Function Documentation

4.25.1.1 void simulationOperations (int num_users, Smartphones smM, Smartphones::iterator it_smM, Events eventsL, Events::iterator it_eventsL, Users usersM, Users::iterator it_usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCityMap::iterator it_grcM, int days)

It creates the SimulationData.txt file useful to compute the statistics and it is the core of the simulator.

4.26 src/Smartphones.cc File Reference

```
#include "../Headers/Smartphones.h"
#include "../Headers/Utilities.h"
```

Functions

- [Smartphones setSmartphones](#) (int num_users)

4.26.1 Function Documentation

4.26.1.1 Smartphones setSmartphones (int num_users)

It sets the details of the smartphone for each user

4.27 src/Statistics.cc File Reference

```
#include "../Headers/Statistics.h"
#include "../Headers/Utilities.h"
#include "../Headers/ReadValues.h"
```

Functions

- void [computeStatistics](#) (int num_users, Smartphones smM, Smartphones::iterator it_smM, Events eventsL, Events::iterator it_eventsL, Users usersM, Users::iterator it_usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCityMap::iterator it_grcM, int days)

4.27.1 Function Documentation

4.27.1.1 void computeStatistics (int num_users, Smartphones smM, Smartphones::iterator it_smM, Events eventsL, Events::iterator it_eventsL, Users usersM, Users::iterator it_usersM, StreetPoint stpM, StreetPoint::iterator it_stpM, GraphCityMap grcM, GraphCityMap::iterator it_grcM, int days)

It computes the statistics parameters from the SimulationData.txt file Delimiter constants for reading file.

4.28 src/User.cc File Reference

```
#include "../Headers/User.h"
#include "../Headers/ReadValues.h"
#include "../Headers/Utilities.h"
```

Functions

- [Users allocateUsers](#) (int num_users, [StreetPoint stpM](#), [StreetPoint::iterator it_stpM](#))
- [Users allocateUsersSlotMobility](#) (int num_users_tot, [StreetPoint stpM](#), [StreetPoint::iterator it_stpM](#))

4.28.1 Function Documentation

4.28.1.1 Users allocateUsers (int num_users, [StreetPoint stpM](#), [StreetPoint::iterator it_stpM](#))

It allocates the users in the map. More in details the user are allocated evenly in the timestamp set from [User](#) It allocates users in the map. The first position is chosen casually.

4.28.1.2 Users allocateUsersSlotMobility (int num_users_tot, [StreetPoint stpM](#), [StreetPoint::iterator it_stpM](#))

It allocates the users in the map. More in details the number of users allocated depends on the probability of presence for every time slot It allocates users in different slots time, according to the presence probability. If it is higher there are many users in that slot. There are 10 slots and each one lasts 1 hour. If it chosen this kind of users allocation, the simulation starts at 21 and finish at 7.

4.29 src/Utilities.cc File Reference

```
#include "../Headers/Utilities.h"
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib>
#include <sstream>
#include <vector>
#include <cmath>
#include <stdlib.h>
#include <map>
#include <list>
#include <algorithm>
#include <iomanip>
#include <time.h>
```

Macros

- [#define pi](#) 3.14159265358979323846
- [#define R](#) 6371
- [#define TO_RAD](#) (3.1415926536 / 180)

Functions

- double `fRand` (double *fMin*, double *fMax*)
- long `rnd32` (long *seed*)
- double `uniform` (double *a*, double *b*, long **seed*)
- double `havdist` (double *th1*, double *ph1*, double *th2*, double *ph2*)
- std::string `PrintByteUnit` (long double *bytes*)
- std::string `PrintByteUnitInMiB` (long double *bytes*)

4.29.1 Macro Definition Documentation

4.29.1.1 `#define pi 3.14159265358979323846`

4.29.1.2 `#define R 6371`

4.29.1.3 `#define TO_RAD (3.1415926536 / 180)`

4.29.2 Function Documentation

4.29.2.1 double `fRand` (double *fMin*, double *fMax*)

4.29.2.2 double `havdist` (double *th1*, double *ph1*, double *th2*, double *ph2*)

It computes the distance in meters between two different points, given the Latitude and Longitude coordinates.

4.29.2.3 std::string `PrintByteUnit` (long double *bytes*)

It converts bytes values in KB/MB/GB automatically.

4.29.2.4 std::string `PrintByteUnitInMiB` (long double *bytes*)

It converts byte values in MiB.

4.29.2.5 long `rnd32` (long *seed*)

4.29.2.6 double `uniform` (double *a*, double *b*, long **seed*)

Index

- allocateUsers
 - ReadValues.h, 18
 - User.cc, 34
 - User.h, 22
- allocateUsersSlotMobility
 - User.cc, 34
 - User.h, 22
- alt
 - Location, 7
 - LocationA, 8
- Antenna, 5
 - Antenna, 5
 - id_antenna, 5
 - loc, 5
 - type_antenna, 5
- Antenna.cc
 - antennasLFunc, 25
 - it_antennasLFunc, 25
 - mapAntenna, 25
- Antenna.h
 - Antennas, 13
 - mapAntenna, 14
- Antennas
 - Antenna.h, 13
- antennasLFunc
 - Antenna.cc, 25
- antennasL
 - CrowdSenSim.cpp, 27
- battery
 - Smartphone, 10
- ClockManagement.cc
 - controlTime, 25
- ClockManagement.h
 - controlTime, 14
- computeStatistics
 - Statistics.cc, 33
 - Statistics.h, 21
- context
 - Smartphone, 10
- controlTime
 - ClockManagement.cc, 25
 - ClockManagement.h, 14
- creatingListOfEvents
 - Event.cc, 28
 - Event.h, 15
- creatingListOfEventsSlots
 - Event.cc, 28
 - Event.h, 15
- creatingMapAssociationGraphPoints
 - MapCity.cc, 30
 - MapCity.h, 17
- CrowdSenSim.cpp
 - antennasL, 27
 - days, 27
 - eventsL, 27
 - grcM, 27
 - it_antennasL, 27
 - it_eventsL, 27
 - it_grcM, 27
 - it_smM, 27
 - it_stpM, 27
 - it_usersM, 27
 - it_usersstatM, 27
 - main, 26
 - num_users, 27
 - smM, 27
 - stpM, 27
 - usersM, 27
 - usersstatM, 27
- DatasentM
 - Smartphones.h, 21
- DatatbdeIM
 - Smartphones.h, 21
- days
 - CrowdSenSim.cpp, 27
- dist
 - MapCity.cc, 30
 - MapCity.h, 17
- dtbd
 - Smartphone, 10
- dts
 - Smartphone, 11
- Event, 6
 - Event, 6
 - id_user, 6
 - loc, 6
 - timestamp, 6
- Event.cc
 - creatingListOfEvents, 28
 - creatingListOfEventsSlots, 28
 - eventComparator, 28
 - eventsLFunc, 28
 - it_eventsLFunc, 28
 - readListOfEvents, 28
- Event.h
 - creatingListOfEvents, 15

- creatingListOfEventsSlots, 15
 - eventComparator, 15
 - Events, 15
 - GraphCityMap, 15
 - readListOfEvents, 15
- eventComparator
 - Event.cc, 28
 - Event.h, 15
- Events
 - Event.h, 15
- eventsLFunc
 - Event.cc, 28
- eventsL
 - CrowdSenSim.cpp, 27
- fRand
 - Utilities.cc, 35
 - Utilities.h, 24
- GraphCityMap
 - Event.h, 15
 - MapCity.h, 17
- grcM
 - CrowdSenSim.cpp, 27
- havdist
 - Utilities.cc, 35
 - Utilities.h, 24
- Headers/Antenna.h, 13
- Headers/ClockManagement.h, 14
- Headers/Event.h, 14
- Headers/HeatMap.h, 16
- Headers/Location.h, 16
- Headers/MapCity.h, 17
- Headers/Position.h, 18
- Headers/ReadValues.h, 18
- Headers/Sample.h, 19
- Headers/Simulation.h, 20
- Headers/Smartphones.h, 20
- Headers/Statistics.h, 21
- Headers/User.h, 22
- Headers/Utilities.h, 23
- HeatMap.cc
 - heatMapAntennas, 29
- HeatMap.h
 - heatMapAntennas, 16
- heatMapAntennas
 - HeatMap.cc, 29
 - HeatMap.h, 16
- id
 - LocationA, 8
- id_antenna
 - Antenna, 5
- id_pos
 - User, 11
- id_smartphone
 - User, 11
- id_user
 - Event, 6
 - Smartphone, 11
- ifsent
 - Sample, 9
- it_antennasLFunc
 - Antenna.cc, 25
- it_antennasL
 - CrowdSenSim.cpp, 27
- it_eventsLFunc
 - Event.cc, 28
- it_eventsL
 - CrowdSenSim.cpp, 27
- it_grcM
 - CrowdSenSim.cpp, 27
- it_locationsLA
 - MapCity.cc, 31
- it_smM
 - CrowdSenSim.cpp, 27
- it_stpM
 - CrowdSenSim.cpp, 27
- it_usersM
 - CrowdSenSim.cpp, 27
- it_usersstatM
 - CrowdSenSim.cpp, 27
- LASTXN
 - Utilities.h, 24
- lat
 - Location, 7
 - LocationA, 8
- loc
 - Antenna, 5
 - Event, 6
 - Position, 8
- Location, 7
 - alt, 7
 - lat, 7
 - Location, 7
 - lon, 7
- Location.h
 - StreetPoint, 17
- LocationA, 7
 - alt, 8
 - id, 8
 - lat, 8
 - LocationA, 8
 - lon, 8
- locationsLA
 - MapCity.cc, 31
- lon
 - Location, 7
 - LocationA, 8
- MODULE
 - Utilities.h, 24
- MYA
 - Utilities.h, 24
- main
 - CrowdSenSim.cpp, 26

- mapAntenna
 - Antenna.cc, [25](#)
 - Antenna.h, [14](#)
- MapCity.cc
 - creatingMapAssociationGraphPoints, [30](#)
 - dist, [30](#)
 - it_locationsLA, [31](#)
 - locationsLA, [31](#)
 - pi, [30](#)
 - R, [30](#)
 - TO_RAD, [30](#)
- MapCity.h
 - creatingMapAssociationGraphPoints, [17](#)
 - dist, [17](#)
 - GraphCityMap, [17](#)
- num_users
 - CrowdSenSim.cpp, [27](#)
- pi
 - MapCity.cc, [30](#)
 - Utilities.cc, [35](#)
 - Utilities.h, [24](#)
- pos
 - Sample, [9](#)
 - User, [12](#)
- Position, [8](#)
 - loc, [8](#)
 - Position, [8](#)
 - timestamp, [8](#)
- PrintByteUnit
 - Utilities.cc, [35](#)
 - Utilities.h, [24](#)
- PrintByteUnitInMiB
 - Utilities.cc, [35](#)
 - Utilities.h, [24](#)
- R
 - MapCity.cc, [30](#)
 - Utilities.cc, [35](#)
 - Utilities.h, [24](#)
- RATIO
 - Utilities.h, [24](#)
- readDaySimulation
 - ReadValues.cc, [31](#)
 - ReadValues.h, [18](#)
- readDecision
 - ReadValues.cc, [31](#)
 - ReadValues.h, [19](#)
- readDefaultStreetPoint
 - ReadValues.cc, [31](#)
 - ReadValues.h, [19](#)
- readFinishMinute
 - ReadValues.cc, [31](#)
 - ReadValues.h, [19](#)
- readFinishtHour
 - ReadValues.cc, [31](#)
 - ReadValues.h, [19](#)
- readKindAntennaFromSetupFile
 - ReadValues.cc, [32](#)
 - ReadValues.h, [19](#)
- readListofEvents
 - Event.cc, [28](#)
 - Event.h, [15](#)
- readMaximumTravelTime
 - ReadValues.cc, [32](#)
 - ReadValues.h, [19](#)
- readMinimumTravelTime
 - ReadValues.cc, [32](#)
 - ReadValues.h, [19](#)
- readNumberUsers
 - ReadValues.cc, [32](#)
 - ReadValues.h, [19](#)
- readRay
 - ReadValues.cc, [32](#)
 - ReadValues.h, [19](#)
- readStartHour
 - ReadValues.cc, [32](#)
 - ReadValues.h, [19](#)
- readStartMinute
 - ReadValues.cc, [32](#)
 - ReadValues.h, [19](#)
- readStreetPoint
 - ReadValues.cc, [32](#)
 - ReadValues.h, [19](#)
- ReadValues.cc
 - readDaySimulation, [31](#)
 - readDecision, [31](#)
 - readDefaultStreetPoint, [31](#)
 - readFinishMinute, [31](#)
 - readFinishtHour, [31](#)
 - readKindAntennaFromSetupFile, [32](#)
 - readMaximumTravelTime, [32](#)
 - readMinimumTravelTime, [32](#)
 - readNumberUsers, [32](#)
 - readRay, [32](#)
 - readStartHour, [32](#)
 - readStartMinute, [32](#)
 - readStreetPoint, [32](#)
 - setDefaultMapCity, [32](#)
 - setMapCity, [32](#)
- ReadValues.h
 - allocateUsers, [18](#)
 - readDaySimulation, [18](#)
 - readDecision, [19](#)
 - readDefaultStreetPoint, [19](#)
 - readFinishMinute, [19](#)
 - readFinishtHour, [19](#)
 - readKindAntennaFromSetupFile, [19](#)
 - readMaximumTravelTime, [19](#)
 - readMinimumTravelTime, [19](#)
 - readNumberUsers, [19](#)
 - readRay, [19](#)
 - readStartHour, [19](#)
 - readStartMinute, [19](#)
 - readStreetPoint, [19](#)
 - setDefaultMapCity, [19](#)

- setMapCity, 19
- rnd32
 - Utilities.cc, 35
 - Utilities.h, 24
- Sample, 9
 - ifsent, 9
 - pos, 9
 - Sample, 9
 - size, 9
 - type, 9
 - value, 10
- Sample.h
 - Samples, 20
- Samples
 - Sample.h, 20
- setDefaultMapCity
 - ReadValues.cc, 32
 - ReadValues.h, 19
- setMapCity
 - ReadValues.cc, 32
 - ReadValues.h, 19
- setSmartphones
 - Smartphones.cc, 33
 - Smartphones.h, 21
- Simulation.cc
 - simulationOperations, 33
- Simulation.h
 - simulationOperations, 20
- simulationOperations
 - Simulation.cc, 33
 - Simulation.h, 20
- size
 - Sample, 9
- Smartphone, 10
 - battery, 10
 - context, 10
 - dtbd, 10
 - dts, 11
 - id_user, 11
 - Smartphone, 10
 - smp, 11
- Smartphones
 - Smartphones.h, 21
- Smartphones.cc
 - setSmartphones, 33
- Smartphones.h
 - DatasentM, 21
 - DatatbdeIM, 21
 - setSmartphones, 21
 - Smartphones, 21
- smM
 - CrowdSenSim.cpp, 27
- smp
 - Smartphone, 11
- src/Antenna.cc, 24
- src/ClockManagement.cc, 25
- src/CrowdSenSim.cpp, 26
- src/Event.cc, 27
- src/HeatMap.cc, 28
- src/Location.cc, 29
- src/MapCity.cc, 29
- src/Position.cc, 31
- src/ReadValues.cc, 31
- src/Sample.cc, 32
- src/Simulation.cc, 32
- src/Smartphones.cc, 33
- src/Statistics.cc, 33
- src/User.cc, 34
- src/Utilities.cc, 34
- Statistics.cc
 - computeStatistics, 33
- Statistics.h
 - computeStatistics, 21
- stpM
 - CrowdSenSim.cpp, 27
- StreetPoint
 - Location.h, 17
- TO_RAD
 - MapCity.cc, 30
 - Utilities.cc, 35
 - Utilities.h, 24
- timestamp
 - Event, 6
 - Position, 8
- tot_data_gen
 - UserStat, 12
- tot_en_tx
 - UserStat, 12
- tot_energy
 - UserStat, 12
- tot_num_samples
 - UserStat, 12
- type
 - Sample, 9
- type_antenna
 - Antenna, 5
- UPTOMOD
 - Utilities.h, 24
- uniform
 - Utilities.cc, 35
 - Utilities.h, 24
- User, 11
 - id_pos, 11
 - id_smartphone, 11
 - pos, 12
 - User, 11
- User.cc
 - allocateUsers, 34
 - allocateUsersSlotMobility, 34
- User.h
 - allocateUsers, 22
 - allocateUsersSlotMobility, 22
 - Users, 22
 - UsersStat, 22
- UserStat, 12

- [tot_data_gen](#), [12](#)
 - [tot_en_tx](#), [12](#)
 - [tot_energy](#), [12](#)
 - [tot_num_samples](#), [12](#)
 - [UserStat](#), [12](#)
- Users
 - [User.h](#), [22](#)
- UsersStat
 - [User.h](#), [22](#)
- usersM
 - [CrowdSenSim.cpp](#), [27](#)
- usersstatM
 - [CrowdSenSim.cpp](#), [27](#)
- Utilities.cc
 - [fRand](#), [35](#)
 - [havdist](#), [35](#)
 - [pi](#), [35](#)
 - [PrintByteUnit](#), [35](#)
 - [PrintByteUnitInMiB](#), [35](#)
 - [R](#), [35](#)
 - [rnd32](#), [35](#)
 - [TO_RAD](#), [35](#)
 - [uniform](#), [35](#)
- Utilities.h
 - [fRand](#), [24](#)
 - [havdist](#), [24](#)
 - [LASTXN](#), [24](#)
 - [MODULE](#), [24](#)
 - [MYA](#), [24](#)
 - [pi](#), [24](#)
 - [PrintByteUnit](#), [24](#)
 - [PrintByteUnitInMiB](#), [24](#)
 - [R](#), [24](#)
 - [RATIO](#), [24](#)
 - [rnd32](#), [24](#)
 - [TO_RAD](#), [24](#)
 - [UPTOMOD](#), [24](#)
 - [uniform](#), [24](#)
- value
 - [Sample](#), [10](#)