ReMoDy Reference Manual

1.0

Generated by Doxygen 1.5.2

Mon May 5 10:02:10 2008

Contents

Chapter 1

ReMoDy Source Code Documentation

1.1 Introduction

This is the documentation of the ReMoDy (Reactive Molecular Dynamics) program, which implements the model of reactive molecular dynamics based on the `Collision Theory`.

1.2 Installation and Execution

1.2.1 Step 1: Extracting the archive

If you have the archive file, like remody.tbz (tar-bzipped) or remody.tgz (tar-gzipped) then files can be retrieved into a current directory as:

tar cvjf remody.tbz

or

tar cvzf remody.tgz

respectively.

1.2.2 Step 2: Compiling

To compile the executable on Linux run make from the remody root directory, where the makefile was saved. Note that the subdirectories src/, run/, and obj/ should be also present.

1.2.3 Step 3: Running

The executables are saved in the run/ directory. It should also contain the example remody.xml and demo.xml files, as well as an initial empty input file empty.dat.gz. To run the program with an OpenGL window active (slow, good for debugging), one can use the command:

./view -f demo.xml empty.dat.gz

This will start the program with the initial parameters read from the demo.xml file and the initial

data read from empty.dat.gz file. On startup the program will open the window. One can point at the window with the mouse and press 'f' key to show the frame and 'r' key to run the simulation. Alternatively, one can use 's' key to run iterations step-by-step. Other key functions are described by pressing the '?' key.

To run the program in a batch mode without OpenGL output, one can use this command:

./job -f demo.xml empty.dat.gz &

This will start the run. To change the parameters of the job, one should modify the input xml file accordingly (demo.xml, remody.xml, etc.).

1.3 Setup

Both view (OpenGL based) and job (batch mode) executables read the configuration xml file. By default the file will have the same name as the executable, for example: job.xml or view.xml. This can be overwritten with the '-f' option. Two example files `remody.xml` and `demo.xml` are provided with the distribution.

The format of the xml file is in most cases self-explanatory with explanations provided in the "title" fields and comments. The main sections include: RUN, CHEMISTRY, DOMAIN, and GUI.

The RUN section specifies the number of iterations, time steps, output intervals, and the maximum number of molecules to be used.

The CHEMISTRY section provides the list of species, and reactions between them.

The DOMAIN section provides the specifications of the boundary conditions and the surface reactions and species.

The GUI section specifies the parameters used in the OpenGL output window, such as colors, line width, etc.

1.4 Licencing and Support

The rest of this documentation provides the descriptioin of the structure and functionality of the classes, namespaces, and files of the code.

The code is provided under the `GPL licence`. Any suggestions/requests can be directed to the developers team at `NIFT`.

Chapter 2

ReMoDy Namespace Index

2.1 ReMoDy Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

ReMoDy Class Index

3.1 ReMoDy Class List

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

Chapter 4

ReMoDy File Index

4.1 ReMoDy File List

Here is a list of all files with brief descriptions:

Chapter 5

ReMoDy Namespace Documentation

5.1 Geom Namespace Reference

Provides some trigeometry functions.

Functions

- REAL **normalize** (REAL a[])
- void **distvec** (REAL a[], REAL b[], REAL c[])
- REAL **distance** (REAL a[], REAL b[])
- REAL **distance** (int dim, REAL a[], REAL b[])
- REAL **sclp** (REAL a[], REAL b[])
- int **hash** (int **i**, int j, int **n**)
- REAL **length** (REAL a[])
- REAL **area** (REAL a[], REAL b[])

5.1.1 Detailed Description

Provides some trigeometry functions.

5.1.2 Function Documentation

5.1.2.1 REAL Geom::area (REAL *a*[], REAL *b*[])

Definition at line 124 of file model.cc.

References DIM, LENGTH, REAL, and VECP.

Referenced by Domain::Domain(), and Domain::injection().

5.1.2.2 REAL Geom::distance (int *dim*, REAL *a*[], REAL *b*[])

Definition at line 100 of file model.cc.

References i, and REAL.

5.1.2.3 REAL Geom::distance (REAL *a*[ ], REAL *b*[ ])

Definition at line 92 of file model.cc.

References DIM, i, and REAL.

5.1.2.4 void Geom::distvec (REAL *a*[ ], REAL *b*[ ], REAL *c*[ ])

Definition at line 89 of file model.cc.

References DIM, and i.

5.1.2.5 int Geom::hash (int *i*, int *j*, int *n*)

Definition at line 113 of file model.cc.

5.1.2.6 REAL Geom::length (REAL *a*[ ])

Definition at line 116 of file model.cc.

References DIM, i, and REAL.

Referenced by normalize().

5.1.2.7 REAL Geom::normalize (REAL *a*[ ])

Definition at line 84 of file model.cc.

References DIM, i, length(), and REAL.

5.1.2.8 REAL Geom::sclp (REAL *a*[ ], REAL *b*[ ])

Definition at line 108 of file model.cc.

References DIM, i, and REAL.

5.2 GridContainer Namespace Reference

Domain (p. 70) Segmentation for Interaction Acceleration.

Functions

- void **init** (REAL ymin[], REAL ymax[], REAL radius, **Pool**< **Molecule** > *newpool)
- int **index** (int ind[])
- void **index** (int **n**, int ind[])
- void **put** (**Molecule** *node)
- **Container**< **Molecule** > * **get** (int ind[])
- int * **dimensions** ()
- bool **checkPool** (int icell, char *msg)

Variables

- REAL **xmin** [DIM]
- REAL **xmax** [DIM]
- REAL **cellsize**
- int **ncells** [DIM+1]
- int **mcells** = 0
- int **mcells1** = 0
- **Container**< **Molecule** > * **nodes**
- **Pool**< **Molecule** > * **pool**
- REAL **cellsize**
- int **ncells** [DIM+1]
- **Container**< **Molecule** > * **nodes**
- **Pool**< **Molecule** > * **pool**

5.2.1 Detailed Description

Domain (p. 70) Segmentation for Interaction Acceleration.

Splits space into cubic cells to consider local interactions between ajacent cells only

5.2.2 Function Documentation

5.2.2.1 bool GridContainer::checkPool (int *icell*, char * *msg*)

5.2.2.2 int * GridContainer::dimensions ()

Definition at line 124 of file grid.cc.

References ncells.

5.2.2.3 Container< Molecule > * GridContainer::get (int *ind*[])

Definition at line 121 of file grid.cc.

References index(), and nodes.

5.2.2.4 void GridContainer::index (int *n*, int *ind*[ ])

Definition at line 67 of file grid.cc.

References DIM, i, mcells1, and ncells.

Referenced by Domain::interaction().

5.2.2.5 int GridContainer::index (int *ind*[ ])

Definition at line 60 of file grid.cc.

References DIM, i, n, and ncells.

Referenced by get(), and put().

5.2.2.6 void GridContainer::init (REAL *ymin*[ ], REAL *ymax*[ ], REAL *radius*, Pool< Molecule > * *newpool*)

Definition at line 24 of file grid.cc.

References cellsize, DIM, i, mcells, mcells1, n, ncells, nodes, Run::option, pool, REAL, xmax, and xmin.

5.2.2.7 void GridContainer::put (Molecule * *node*)

DDD

Definition at line 80 of file grid.cc.

References cellsize, Molecule::Coordinates(), DIM, i, index(), ncells, nodes, REAL, Molecule::Type(), VOIDSPECIE, and xmin.

Referenced by Boundary::Inject(), and Domain::run().

5.2.3 Variable Documentation

5.2.3.1 REAL GridContainer::cellsize

Definition at line 19 of file grid.cc.

Referenced by Domain::Domain(), init(), and put().

5.2.3.2 REAL GridContainer::cellsize

Definition at line 19 of file grid.cc.

Referenced by Domain::Domain(), init(), and put().

5.2.3.3 int GridContainer::mcells = 0

Definition at line 21 of file grid.cc.

Referenced by init().

5.2.3.4 int GridContainer::mcells1 = 0

Definition at line 21 of file grid.cc.

Referenced by index(), and init().

5.2.3.5 int GridContainer::ncells[DIM+1]

Definition at line 20 of file grid.cc.

Referenced by dimensions(), index(), init(), Domain::interaction(), and put().

5.2.3.6 int GridContainer::ncells[DIM+1]

Definition at line 20 of file grid.cc.

Referenced by dimensions(), index(), init(), Domain::interaction(), and put().

5.2.3.7 Container<Molecule>* GridContainer::nodes

Definition at line 22 of file grid.cc.

Referenced by Domain::boundary(), get(), init(), and put().

5.2.3.8 Container<Molecule>* GridContainer::nodes

Definition at line 22 of file grid.cc.

Referenced by Domain::boundary(), get(), init(), and put().

5.2.3.9 Pool<Molecule>* GridContainer::pool

Definition at line 23 of file grid.cc.

Referenced by Container< Element >::append(), Container< Element >::checkPool(), init(), Container< Element >::insert(), and Container< Element >::remove().

5.2.3.10 Pool<Molecule>* GridContainer::pool

Definition at line 23 of file grid.cc.

Referenced by Container< Element >::append(), Container< Element >::checkPool(), init(), Container< Element >::insert(), and Container< Element >::remove().

5.2.3.11 REAL GridContainer::xmax[DIM]

Definition at line 18 of file grid.cc.

Referenced by init().

5.2.3.12 REAL GridContainer::xmin[DIM]

Definition at line 18 of file grid.cc.

Referenced by init(), and put().

5.3 Gui Namespace Reference

OpenGL window output routines.

Classes

- struct **ElementDisp**
- struct **ColorScale**
- struct **Scene**
- struct **WindowGeom**

Enumerations

- enum **Elements** {
  points = 0, **nodes**, **edges**, **faces**,
  cells, **frame**, **mesh**, **boundary_nodes**,
  boundary_edges, **boundary_faces**, **boundary_cells**, **maxelements** }
- enum **Color** {
  red = 0, **green**, **blue**, **skyblue**,
  brown, **magenta**, **yellow**, **color6**,
  color7, **color8**, **color9**, **color10**,
  color11, **color12**, **color13**, **color14**,
  color15, **maxcolor** }
- enum **ColorSchemes** {
  colorByType = 0, **colorByBoundary**, **colorByTime**, **colorByMass**,
  maxColorSchemes }
- enum **PointParameters** {
  variable = maxcolor, **vmin**, **vmax**, **size**,
  sizevar, **massvar**, **maxpointprm** }
- enum **LineParameters** { **thickness** = maxcolor, **length**, **maxlineprm** }
- enum **AxesParameters** {
  axeswidth = 0, **xaxislength**, **yaxislength**, **zaxislength**,
  arrowheight, **arrowidth**, **maxaxesprm** }
- enum **SurfDispType** { **gridlines** = 0, **solidsurface**, **maxsurfdisptypes** }
- enum **Movement** { **stay** = 0, **rotate**, **moveuv**, **movew** }
- enum **ShowPars** {
  showRun = 0, **showAxes**, **showSpheres**, **showBonds**,
  showGrid, **showNodes**, **showVariables**, **showBoundaryVertexes**,
  showBoundaryVectors, **showToolVertexes**, **showBoundaryFaceCenters**, **showBoundaryFaces**,
  showBoundaryGrid, **showToolGrid**, **showFrame**, **showCellCenters**,
  showFaceCenters, **showIsoSurfaces**, **dumpWindow**, **maxshowpars** }

Functions

- int **query_extension** (char *extName)
- void **init** (int argc, char *argv[], **Domain** *newdomain)
- void **helpDisplay** ()
- int **readparam** (char *s, char *param[], int maxparam, char *filename, int val[])
- int **readparam** (char *s, char *param[], int maxparam, char *filename, REAL val[])
- void **readconf** ()
- void **refresh** ()
- void **initdisp** ()
- void **Materials** (int argc, char *argv[])
- void **showVector** (double *x, double *v)
- void **showParticle** (double vmn, double vmx, double val, double *x)
- void **showSphere** (double vmn, double vmx, double val, double rad, double *x)
- void **displayAxes** ()
- void **getScaling** (double &l0, double &l1)
- void **getXLimits** (double *x0, double *x1)
- void **Exit** ()
- void **Quit** ()
- void **menu** (int value)
- void **displayMenu** ()
- void **consoleMenu** ()
- void **setBackgroundRun** ()
- void **setForegroundRun** ()
- void **toggleSpheres** ()
- void **switchColorScheme** ()
- void **runmany** ()
- void **toggleWindowDump** ()
- void **dumpwindow** ()
- void **commandMode** ()
- void **display** ()
- void **animate** ()
- void **helpCommand** ()
- void **reshape** (int w, int h)
- void **mouse** (int button, int state, int x, int **y**)
- void **motion** (int x, int **y**)
- void **keyboard** (unsigned int key)
- void **run** ()
- void **displaymessage** (char *msg)
- void **setElementColor** (int element_type, REAL **rgbcolor**[])
- void **getElementColor** (int element_type, REAL **rgbcolor**[])
- void **writeData** ()
- void **printGridXLimits** ()
- void **printGridVecLimits** ()
- void **getXLimits** (REAL ***xmin**, REAL ***xmax**)
- void **getScaling** (REAL &**lmin**, REAL &**lmax**)
- void **printPartVelLimits** ()
- void **printParticleVariables** ()
- void **showGridElements** (int ne, double *X, REAL color[])
- void **showVectorComponent** (int ivar, int icomp, double *X, double vmn, double vmx)

- void **showVector** (int ivar, double *X)
- void **printVariables** (int **n**)
- void **finish** ()
- void **InitMaterials** (void)
- void **selectVariable** ()
- void **keyboard** (unsigned char key, int x, int **y**)
- void **drawSegment** (REAL *x, REAL *y)

Variables

- **Domain * domain**
- **ElementDisp disp** [maxelements]
- char * **configfile** = "gui.cfg"
- char **windowname** [MAXLINLEN]
- int **showpar** [maxshowpars]
- int **finished**
- int **animation**
- int **nwdump**
- int **iwdump**
- int **attributeList** [ ]
- REAL **step**
- REAL **vecval** [maxlineprm]
- REAL **axes** [maxaxesprm]
- REAL **rgbcolor** [maxcolor][3]
- REAL **wdtime**
- REAL **xo** [3]
- REAL **dx**
- REAL **dy**
- REAL **dz**
- REAL **lastx**
- REAL **lasty**
- REAL **lastz**
- int **mouseButtons** [3]
- REAL **zoom**
- REAL **rotx**
- REAL **roty**
- REAL **tx**
- REAL **ty**
- REAL **xmin** [DIM]
- REAL **xmax** [DIM]
- REAL **lx**
- REAL **ly**
- REAL **lz**
- REAL **lmin**
- REAL **lmax**
- enum **Movement movement**
- **ColorScale colorscale**
- **Scene scene**
- **WindowGeom window**
- BFaceList * **bface_root**

- Display * **dpy**
- Window **win**
- int **firstR** = 1
- int * **vars**
- int * **coms**
- char * **configfile**
- REAL **lx**
- REAL **ly**
- REAL **lz**
- REAL **lmin**
- REAL **lmax**
- REAL **step**
- REAL **vecval** [maxlineprm]
- REAL **axes** [maxaxesprm]
- REAL **rgbcolor** [maxcolor][3]
- **ElementDisp disp** [maxelements]
- int **showpar** [maxshowpars]
- int **finished**
- int **animation**
- REAL **dx**
- REAL **dy**
- REAL **dz**
- REAL **lastx**
- REAL **lasty**
- REAL **lastz**
- int **mouseButtons** [3]
- REAL **zoom**
- REAL **rotx**
- REAL **roty**
- REAL **tx**
- REAL **ty**

5.3.1 Detailed Description

OpenGL window output routines.

5.3.2 Enumeration Type Documentation

5.3.2.1 enum Gui::AxesParameters

Enumerator:

axeswidth
xaxislength
yaxislength
zaxislength
arrowheight
arrowidth
maxaxesprm

Definition at line 71 of file gui.h.

5.3.2.2 enum Gui::Color

Enumerator:

red
green
blue
skyblue
brown
magenta
yellow
color6
color7
color8
color9
color10
color11
color12
color13
color14
color15
maxcolor

Definition at line 31 of file gui.h.

5.3.2.3 enum Gui::ColorSchemes

Enumerator:

colorByType
colorByBoundary
colorByTime
colorByMass
maxColorSchemes

Definition at line 51 of file gui.h.

5.3.2.4 enum Gui::Elements

Enumerator:

points
nodes
edges
faces
cells

frame

mesh

boundary_nodes

boundary_edges

boundary_faces

boundary_cells

maxelements

Definition at line 17 of file gui.h.

5.3.2.5 enum Gui::LineParameters

Enumerator:

thickness

length

maxlineprm

Definition at line 65 of file gui.h.

5.3.2.6 enum Gui::Movement

Enumerator:

stay

rotate

moveuv

movew

Definition at line 123 of file gui.h.

5.3.2.7 enum Gui::PointParameters

Enumerator:

variable

vmin

vmax

size

sizevar

massvar

maxpointprm

Definition at line 58 of file gui.h.

5.3.2.8 enum Gui::ShowPars

Enumerator:

showRun
showAxes
showSpheres
showBonds
showGrid
showNodes
showVariables
showBoundaryVertexes
showBoundaryVectors
showToolVertexes
showBoundaryFaceCenters
showBoundaryFaces
showBoundaryGrid
showToolGrid
showFrame
showCellCenters
showFaceCenters
showIsoSurfaces
dumpWindow
maxshowpars

Definition at line 129 of file gui.h.

5.3.2.9 enum Gui::SurfDispType

Enumerator:

gridlines
solidsurface
maxsurfdisptypes

Definition at line 81 of file gui.h.

5.3.3 Function Documentation

5.3.3.1 void Gui::animate ()

DDD glutIdleFunc(NULL);//Causes segmentation fault (!?)

refresh() (p. 26);

Definition at line 1851 of file gui.cc.

References animation, display(), domain, dumpwindow(), dumpWindow, ESC, finished, iwdump, nwdump, Run::option, Domain::run(), showpar, showRun, and Run::time.

Referenced by menu().

5.3.3.2 void Gui::commandMode ()

Definition at line 1465 of file gui.cc.

References consoleMenu(), dumpwindow(), Exit(), helpCommand(), MAXLINLEN, Quit(), runmany(), setBackgroundRun(), setForegroundRun(), switchColorScheme(), toggleSpheres(), and toggleWindowDump().

Referenced by keyboard().

5.3.3.3 void Gui::consoleMenu ()

Definition at line 1360 of file gui.cc.

References displayMenu(), MAXLINLEN, and menu().

Referenced by commandMode(), and keyboard().

5.3.3.4 void Gui::display ()

Definition at line 1530 of file gui.cc.

References blue, Gui::Scene::color, colorByBoundary, colorByMass, colorByTime, colorByType, Molecule::Coordinates(), displayAxes(), domain, dpy, faces, Gui::Scene::frame, frame, getElementColor(), green, i, Palette::init(), Potential::lengthscale(), Gui::Scene::Frame::line, Species::Specie::Mass(), Gui::Scene::Color::maxvalue, Gui::Scene::mesh, Gui::Scene::Color::minvalue, Domain::Molecules(), Gui::Scene::Mesh::node, nodes, Collection< Element >::number(), Run::option, Palette::pickcolor(), REAL, red, refresh(), rotx, roty, scene, Gui::Scene::Color::scheme, showBoundaryGrid, showBoundaryVertexes, showFrame, showGrid, showNodes, showpar, showSpheres, Species::Specie::Size(), size, Species::species, thickness, Run::time, tx, ty, Molecule::Type(), win, xmax, xmin, and zoom.

Referenced by animate().

5.3.3.5 void Gui::displayAxes ()

Definition at line 1181 of file gui.cc.

References arrowheight, arrowidth, axes, axeswidth, lx, ly, lz, showAxes, showpar, xaxislength, xo, yaxislength, and zaxislength.

Referenced by display().

5.3.3.6 void Gui::displayMenu ()

Definition at line 1334 of file gui.cc.

Referenced by consoleMenu().

5.3.3.7 void Gui::displaymessage (char * *msg*)

Definition at line 2162 of file gui.cc.

5.3.3.8 void Gui::drawSegment (REAL * *x*, REAL * *y*)

5.3.3.9 void Gui::dumpwindow ()

Definition at line 1461 of file gui.cc.

References windowname, and IO::xwd().

Referenced by animate(), and commandMode().

5.3.3.10 void Gui::Exit ()

Definition at line 1229 of file gui.cc.

References animation, and MAXLINLEN.

Referenced by commandMode(), and keyboard().

5.3.3.11 void Gui::finish ()

5.3.3.12 void Gui::getElementColor (int *element_type*, REAL *rgbcolor*[])

Definition at line 2172 of file gui.cc.

References disp, and maxelements.

Referenced by display().

5.3.3.13 void Gui::getScaling (REAL & *lmin*, REAL & *lmax*)

Referenced by reshape().

5.3.3.14 void Gui::getScaling (double & *l0*, double & *l1*)

Definition at line 1212 of file gui.cc.

References lmax, and lmin.

5.3.3.15 void Gui::getXLimits (REAL * *xmin*, REAL * *xmax*)

5.3.3.16 void Gui::getXLimits (double * *x0*, double * *x1*)

Definition at line 1220 of file gui.cc.

References i, xmax, and xmin.

5.3.3.17 void Gui::helpCommand ()

Definition at line 1871 of file gui.cc.

References Run::outputname.

Referenced by commandMode().

5.3.3.18 void Gui::helpDisplay ()

Definition at line 175 of file gui.cc.

References configfile, and Run::outputname.

Referenced by init(), keyboard(), and menu().

5.3.3.19 void Gui::init (int *argc*, char * *argv*[], Domain * *newdomain*)

Definition at line 91 of file gui.cc.

References animation, attributeList, Gui::Scene::color, colorByMass, domain, dpy, dumpWindow, ERROR, finished, Gui::WindowGeom::height, helpDisplay(), initdisp(), lastx, lasty, lastz, Materials(), mouseButtons, Run::option, query_extension(), rotx, roty, scene, Gui::Scene::Color::scheme, showAxes, showBonds, showBoundaryFaceCenters, showBoundaryFaces, showBoundaryGrid, showBoundaryVectors, showBoundaryVertexes, showCellCenters, showFaceCenters, showFrame, showGrid, showNodes, showpar, showRun, showSpheres, showToolGrid, showToolVertexes, showVariables, tx, ty, wdtime, Gui::WindowGeom::width, win, window, windowname, and zoom.

Referenced by main().

5.3.3.20 void Gui::initdisp ()

DDD

Definition at line 701 of file gui.cc.

References arrowheight, arrowidth, axes, axeswidth, blue, brown, cells, colorscale, DIM, domain, dx, dy, dz, edges, faces, frame, Gui::Scene::frame, green, Gui::WindowGeom::height, i, LARGE, length, Gui::Scene::Mesh::line, Gui::Scene::Frame::line, lmax, lmin, lx, ly, lz, magenta, MAX_WINDOW_HEIGHT, MAX_WINDOW_WIDTH, maxcolor, maxelements, mesh, Gui::Scene::mesh, Domain::Molecules(), Gui::Scene::Mesh::node, nodes, Collection< Element >::number(), Run::option, readconf(), REAL, red, Gui::ColorScale::relative, rgbcolor, scene, setElementColor(), Domain::setMaxBound(), Domain::setMinBound(), size, skyblue, step, thickness, vecval, Gui::WindowGeom::width, window, WINDOW_SIZE, xaxislength, xmax, xmin, xo, yaxislength, yellow, zaxislength, and zoom.

Referenced by init().

5.3.3.21 void Gui::InitMaterials (void)

5.3.3.22 void Gui::keyboard (unsigned char *key*, int *x*, int *y*)

5.3.3.23 void Gui::keyboard (unsigned int *key*)

refresh() (p. 26);

Definition at line 1978 of file gui.cc.

References animation, commandMode(), consoleMenu(), domain, ESC, Exit(), finished, helpDisplay(), MAXLINLEN, Run::option, Run::outputname, readconf(), Domain::run(), Domain::save(), showAxes, showBoundaryFaces, showBoundaryGrid, showBoundaryVectors, showBoundaryVertexes, showFrame, showGrid, showNodes, showpar, switchColorScheme(), Run::time, and zoom.

5.3.3.24 void Gui::Materials (int *argc*, char * *argv*[ ])

Definition at line 818 of file gui.cc.

References i, lz, Run::option, and xo.

Referenced by init().

5.3.3.25 void Gui::menu (int *value*)

Definition at line 1253 of file gui.cc.

References ABOUT, animate(), animation, finished, helpDisplay(), showAxes, showBonds, showBoundaryFaceCenters, showBoundaryGrid, showBoundaryVectors, showBoundaryVertexes, showCellCenters, showFaceCenters, showGrid, showNodes, showpar, and showVariables.

Referenced by consoleMenu().

5.3.3.26 void Gui::motion (int *x*, int *y*)

Definition at line 1954 of file gui.cc.

References lastx, lasty, mouseButtons, REAL, rotx, roty, tx, ty, and zoom.

5.3.3.27 void Gui::mouse (int *button*, int *state*, int *x*, int *y*)

Definition at line 1927 of file gui.cc.

References lastx, lasty, LEFT_BUTTON, MIDDLE_BUTTON, mouseButtons, and RIGHT_BUTTON.

5.3.3.28 void Gui::printGridVecLimits ()

5.3.3.29 void Gui::printGridXLimits ()

5.3.3.30 void Gui::printParticleVariables ()

5.3.3.31 void Gui::printPartVelLimits ()

5.3.3.32 void Gui::printVariables (int *n*)

5.3.3.33 int Gui::query_extension (char * *extName*)

Definition at line 78 of file gui.cc.

References n.

Referenced by init().

5.3.3.34 void Gui::Quit ()

Definition at line 1241 of file gui.cc.

References animation, and MAXLINLEN.

Referenced by commandMode().

5.3.3.35 void Gui::readconf ()

Definition at line 259 of file gui.cc.

References arrowheight, arrowidth, axeswidth, blue, brown, Gui::Scene::color, colorByMass, colorByType, Run::configfile, configfile, DIM, DOCTYPE, domain, dx, dy, dz, frame, Gui::Scene::frame, green, i, length, Gui::Scene::Frame::line, Gui::Scene::Mesh::line, lx, ly, lz, magenta, massvar, maxaxesprm, Domain::maxBound(), maxcolor, maxlineprm, MAXLINLEN, maxpointprm, maxshowpars, Gui::Scene::Color::maxvalue, mesh, Gui::Scene::mesh, Domain::minBound(), Gui::Scene::Color::minvalue, Gui::Scene::Mesh::node, nodes, Run::option, IO::parseWord(), REAL, red, rgbcolor, scene, Gui::Scene::Color::scheme, setElementColor(), showpar, showSpheres, size, sizevar, skyblue, step, thickness, variable, vecval, vmax, vmin, xaxislength, xmax, xmin, yaxislength, and zaxislength.

Referenced by initdisp(), and keyboard().

5.3.3.36 int Gui::readparam (char * *s*, char * *param*[ ], int *maxparam*, char * *filename*, REAL *val*[ ])

Definition at line 230 of file gui.cc.

References i.

5.3.3.37 int Gui::readparam (char * *s*, char * *param*[ ], int *maxparam*, char * *filename*, int *val*[ ])

Definition at line 198 of file gui.cc.

References i.

5.3.3.38 void Gui::refresh ()

Definition at line 689 of file gui.cc.

Referenced by display().

5.3.3.39 void Gui::reshape (int *w*, int *h*)

Definition at line 1890 of file gui.cc.

References getScaling(), lmax, and lmin.

5.3.3.40 void Gui::run ()

Definition at line 2159 of file gui.cc.

Referenced by main().

5.3.3.41 void Gui::runmany ()

Definition at line 1392 of file gui.cc.

References domain, n, and Domain::run().

Referenced by commandMode().

5.3.3.42 void Gui::selectVariable ()

5.3.3.43 void Gui::setBackgroundRun ()

Definition at line 1376 of file gui.cc.

References showpar, and showRun.

Referenced by commandMode().

5.3.3.44 void Gui::setElementColor (int *element_type*, REAL *rgbcolor*[])

Definition at line 2166 of file gui.cc.

References disp, and maxelements.

Referenced by initdisp(), and readconf().

5.3.3.45 void Gui::setForegroundRun ()

Definition at line 1380 of file gui.cc.

References showpar, and showRun.

Referenced by commandMode().

5.3.3.46 void Gui::showGridElements (int *ne*, double * *X*, REAL *color*[])

5.3.3.47 void Gui::showParticle (double *vmn*, double *vmx*, double *val*, double * *x*)

Definition at line 952 of file gui.cc.

References PI, REAL, and Palette::vav.

5.3.3.48 void Gui::showSphere (double *vmn*, double *vmx*, double *val*, double *rad*, double * *x*)

= { 0.6, 0.8, 0.3, 1.0 },

Definition at line 982 of file gui.cc.

References PI, REAL, and Palette::vav.

5.3.3.49 void Gui::showVector (int *ivar*, double * *X*)

5.3.3.50 void Gui::showVector (double * *x*, double * *v*)

Definition at line 874 of file gui.cc.

References length, and vecval.

5.3.3.51 void Gui::showVectorComponent (int *ivar*, int *icomp*, double * *X*, double *vmn*, double *vmx*)

5.3.3.52 void Gui::switchColorScheme ()

Definition at line 1388 of file gui.cc.

References Gui::Scene::color, maxColorSchemes, scene, and Gui::Scene::Color::scheme.

Referenced by commandMode(), and keyboard().

5.3.3.53 void Gui::toggleSpheres ()

Definition at line 1384 of file gui.cc.

References showpar, and showSpheres.

Referenced by commandMode().

5.3.3.54 void Gui::toggleWindowDump ()

Definition at line 1448 of file gui.cc.

References dumpWindow, iwdump, nwdump, showpar, step, Run::time, and wdtime.

Referenced by commandMode().

5.3.3.55 void Gui::writeData ()

5.3.4 Variable Documentation

5.3.4.1 int Gui::animation

Definition at line 29 of file gui.cc.

Referenced by animate(), Exit(), init(), keyboard(), menu(), and Quit().

5.3.4.2 int Gui::animation

Definition at line 29 of file gui.cc.

Referenced by animate(), Exit(), init(), keyboard(), menu(), and Quit().

5.3.4.3 int Gui::attributeList[ ]

Initial value:

```
{       GLX_RGBA,
        GLX_RED_SIZE, 1,
        GLX_GREEN_SIZE, 1,
        GLX_BLUE_SIZE, 1,
        GLX_DOUBLEBUFFER,
        GLX_DEPTH_SIZE, 1,
        None
}
```

Definition at line 33 of file gui.cc.

Referenced by init().

5.3.4.4 REAL Gui::axes[maxaxesprm]

Definition at line 43 of file gui.cc.

Referenced by displayAxes(), and initdisp().

5.3.4.5 REAL Gui::axes[maxaxesprm]

Definition at line 43 of file gui.cc.

Referenced by displayAxes(), and initdisp().

5.3.4.6 struct BFaceList* Gui::bface_root

Definition at line 70 of file gui.cc.

5.3.4.7 struct ColorScale Gui::colorscale

Definition at line 66 of file gui.cc.

Referenced by initdisp().

5.3.4.8 int * Gui::coms

Definition at line 76 of file gui.cc.

5.3.4.9 char* Gui::configfile

Definition at line 26 of file gui.cc.

Referenced by Domain::Domain(), Run::init(), main(), Run::readcmdline(), and readconf().

5.3.4.10 char* Gui::configfile = "gui.cfg"

Definition at line 26 of file gui.cc.

Referenced by helpDisplay(), and readconf().

5.3.4.11 ElementDisp Gui::disp[maxelements]

Definition at line 25 of file gui.cc.

Referenced by getElementColor(), and setElementColor().

5.3.4.12 ElementDisp Gui::disp[maxelements]

Definition at line 25 of file gui.cc.

Referenced by getElementColor(), and setElementColor().

5.3.4.13 Domain* Gui::domain

Definition at line 24 of file gui.cc.

Referenced by animate(), display(), init(), initdisp(), keyboard(), main(), readconf(), and runmany().

5.3.4.14 Display* Gui::dpy

Definition at line 72 of file gui.cc.

Referenced by display(), and init().

5.3.4.15 REAL Gui::dx

Definition at line 43 of file gui.cc.

Referenced by Domain::boundary(), initdisp(), Domain::interaction(), and readconf().

5.3.4.16 REAL Gui::dx

Definition at line 43 of file gui.cc.

Referenced by Domain::boundary(), initdisp(), Domain::interaction(), and readconf().

5.3.4.17 REAL Gui::dy

Definition at line 43 of file gui.cc.

Referenced by initdisp(), and readconf().

5.3.4.18 REAL Gui::dy

Definition at line 43 of file gui.cc.

Referenced by initdisp(), and readconf().

5.3.4.19 REAL Gui::dz

Definition at line 43 of file gui.cc.

Referenced by initdisp(), and readconf().

5.3.4.20 REAL Gui::dz

Definition at line 43 of file gui.cc.

Referenced by initdisp(), and readconf().

5.3.4.21 int Gui::finished

Definition at line 29 of file gui.cc.

Referenced by animate(), init(), keyboard(), and menu().

5.3.4.22 int Gui::finished

Definition at line 29 of file gui.cc.

Referenced by animate(), init(), keyboard(), and menu().

5.3.4.23 int Gui::firstR = 1

Definition at line 75 of file gui.cc.

5.3.4.24 int Gui::iwdump

Definition at line 29 of file gui.cc.

Referenced by animate(), and toggleWindowDump().

5.3.4.25 REAL Gui::lastx

Definition at line 43 of file gui.cc.

Referenced by init(), motion(), and mouse().

5.3.4.26 REAL Gui::lastx

Definition at line 43 of file gui.cc.

Referenced by init(), motion(), and mouse().

5.3.4.27 REAL Gui::lasty

Definition at line 43 of file gui.cc.

Referenced by init(), motion(), and mouse().

5.3.4.28 REAL Gui::lasty

Definition at line 43 of file gui.cc.

Referenced by init(), motion(), and mouse().

5.3.4.29 REAL Gui::lastz

Definition at line 43 of file gui.cc.

Referenced by init().

5.3.4.30 REAL Gui::lastz

Definition at line 43 of file gui.cc.

Referenced by init().

5.3.4.31 REAL Gui::lmax

Definition at line 61 of file gui.cc.

Referenced by getScaling(), initdisp(), and reshape().

5.3.4.32 REAL Gui::lmax

Definition at line 61 of file gui.cc.

Referenced by getScaling(), initdisp(), and reshape().

5.3.4.33 REAL Gui::lmin

Definition at line 61 of file gui.cc.

Referenced by getScaling(), initdisp(), and reshape().

5.3.4.34 REAL Gui::lmin

Definition at line 61 of file gui.cc.

Referenced by getScaling(), initdisp(), and reshape().

5.3.4.35 REAL Gui::lx

Definition at line 61 of file gui.cc.

Referenced by displayAxes(), initdisp(), and readconf().

5.3.4.36 REAL Gui::lx

Definition at line 61 of file gui.cc.

Referenced by displayAxes(), initdisp(), and readconf().

5.3.4.37 REAL Gui::ly

Definition at line 61 of file gui.cc.

Referenced by displayAxes(), initdisp(), and readconf().

5.3.4.38 REAL Gui::ly

Definition at line 61 of file gui.cc.

Referenced by displayAxes(), initdisp(), and readconf().

5.3.4.39 REAL Gui::lz

Definition at line 61 of file gui.cc.

Referenced by displayAxes(), initdisp(), Materials(), and readconf().

5.3.4.40 REAL Gui::lz

Definition at line 61 of file gui.cc.

Referenced by displayAxes(), initdisp(), Materials(), and readconf().

5.3.4.41 int Gui::mouseButtons[3]

Definition at line 53 of file gui.cc.

Referenced by init(), motion(), and mouse().

5.3.4.42 int Gui::mouseButtons[3]

Definition at line 53 of file gui.cc.

Referenced by init(), motion(), and mouse().

5.3.4.43 enum Movement Gui::movement

Definition at line 64 of file gui.cc.

5.3.4.44 int Gui::nwdump

Definition at line 29 of file gui.cc.

Referenced by animate(), and toggleWindowDump().

5.3.4.45 REAL Gui::rgbcolor[maxcolor][3]

Definition at line 43 of file gui.cc.

Referenced by initdisp(), and readconf().

5.3.4.46 REAL Gui::rgbcolor[maxcolor][3]

Definition at line 43 of file gui.cc.

Referenced by initdisp(), and readconf().

5.3.4.47 REAL Gui::rotx

Definition at line 55 of file gui.cc.

Referenced by display(), init(), and motion().

5.3.4.48 REAL Gui::rotx

Definition at line 55 of file gui.cc.

Referenced by display(), init(), and motion().

5.3.4.49 REAL Gui::roty

Definition at line 56 of file gui.cc.

Referenced by display(), init(), and motion().

5.3.4.50 REAL Gui::roty

Definition at line 56 of file gui.cc.

Referenced by display(), init(), and motion().

5.3.4.51 struct Scene Gui::scene

Definition at line 67 of file gui.cc.

Referenced by display(), init(), initdisp(), readconf(), and switchColorScheme().

5.3.4.52 int Gui::showpar[maxshowpars]

Definition at line 29 of file gui.cc.

Referenced by animate(), display(), displayAxes(), init(), keyboard(), menu(), readconf(), setBackgroundRun(), setForegroundRun(), toggleSpheres(), and toggleWindowDump().

5.3.4.53 int Gui::showpar[maxshowpars]

Definition at line 29 of file gui.cc.

Referenced by animate(), display(), displayAxes(), init(), keyboard(), menu(), readconf(), setBackgroundRun(), setForegroundRun(), toggleSpheres(), and toggleWindowDump().

5.3.4.54 REAL Gui::step

Definition at line 43 of file gui.cc.

Referenced by initdisp(), Domain::injection(), readconf(), Domain::run(), and toggleWindowDump().

5.3.4.55 REAL Gui::step

Definition at line 43 of file gui.cc.

Referenced by initdisp(), Domain::injection(), readconf(), Domain::run(), and toggleWindowDump().

5.3.4.56 REAL Gui::tx

Definition at line 57 of file gui.cc.

Referenced by display(), init(), and motion().

5.3.4.57 REAL Gui::tx

Definition at line 57 of file gui.cc.

Referenced by display(), init(), and motion().

5.3.4.58 REAL Gui::ty

Definition at line 58 of file gui.cc.

Referenced by display(), init(), and motion().

5.3.4.59 REAL Gui::ty

Definition at line 58 of file gui.cc.

Referenced by display(), init(), and motion().

5.3.4.60 int* Gui::vars

Definition at line 76 of file gui.cc.

5.3.4.61 REAL Gui::vecval[maxlineprm]

Definition at line 43 of file gui.cc.

Referenced by initdisp(), readconf(), and showVector().

5.3.4.62 REAL Gui::vecval[maxlineprm]

Definition at line 43 of file gui.cc.

Referenced by initdisp(), readconf(), and showVector().

5.3.4.63 REAL Gui::wdtime

Definition at line 43 of file gui.cc.

Referenced by init(), and toggleWindowDump().

5.3.4.64 Window Gui::win

Definition at line 73 of file gui.cc.

Referenced by display(), and init().

5.3.4.65 struct WindowGeom Gui::window

Definition at line 68 of file gui.cc.

Referenced by init(), and initdisp().

5.3.4.66 char Gui::windowname[MAXLINLEN]

Definition at line 26 of file gui.cc.

Referenced by dumpwindow(), and init().

5.3.4.67 REAL Gui::xmax

Definition at line 61 of file gui.cc.

Referenced by display(), getXLimits(), initdisp(), and readconf().

5.3.4.68 REAL Gui::xmin

Definition at line 61 of file gui.cc.

Referenced by display(), getXLimits(), initdisp(), and readconf().

5.3.4.69 REAL Gui::xo

Definition at line 43 of file gui.cc.

Referenced by displayAxes(), initdisp(), and Materials().

5.3.4.70 REAL Gui::zoom

Definition at line 54 of file gui.cc.

Referenced by display(), init(), initdisp(), keyboard(), and motion().

5.3.4.71 REAL Gui::zoom

Definition at line 54 of file gui.cc.

Referenced by display(), init(), initdisp(), keyboard(), and motion().

5.4 IO Namespace Reference

Some **IO** (p. 37) routines.

Functions

- void **initxwd** (int **i**)
- void **xwd** (char *windowname)
- int **getCharAttr** (xmlNodePtr cur, char *attr, char *result)
- int **getIntAttr** (xmlNodePtr cur, char *attr)
- int **getIntAttr** (xmlDocPtr doc, char *tag, char *keyname, char *key, char *attr)
- int **parseWord** (xmlDocPtr doc, xmlNodePtr cur, char *keyword, char *result)
- int **parseInt** (xmlDocPtr doc, xmlNodePtr cur, char *keyword)
- int **parseFloat** (xmlDocPtr doc, xmlNodePtr cur, char *keyword, REAL &result)
- double **parseFloat** (xmlDocPtr doc, xmlNodePtr cur, char *keyword)
- void **getTimeXML** (char inpfilename[])
- void **getTime** (char inpfilename[])
- int **getIter** (char inpfilename[])

Variables

- int **ioutput** = 0
- int **nxwdump** = 0
- int **ioutput**

5.4.1 Detailed Description

Some **IO** (p. 37) routines.

5.4.2 Function Documentation

5.4.2.1 int IO::getCharAttr (xmlNodePtr *cur*, char * *attr*, char * *result*)

Definition at line 40 of file io.cc.

References MAXLINLEN, and Run::option.

Referenced by Domain::Domain(), and getIntAttr().

5.4.2.2 int IO::getIntAttr (xmlDocPtr *doc*, char * *tag*, char * *keyname*, char * *key*, char * *attr*)

xmlChar *key;

val=getIntAttr(cur,attr);

Definition at line 77 of file io.cc.

References getCharAttr(), MAXLINLEN, Run::option, parseInt(), and Potential::value().

5.4.2.3 int IO::getIntAttr (xmlNodePtr *cur*, char * *attr*)

Definition at line 57 of file io.cc.

References Run::option.

5.4.2.4 int IO::getIter (char *inpfilename*[])

Definition at line 249 of file io.cc.

References DOCTYPE, MAXLINLEN, and Run::option.

Referenced by main().

5.4.2.5 void IO::getTime (char *inpfilename*[])

Definition at line 239 of file io.cc.

References getTimeXML(), Run::option, and Run::time.

Referenced by Run::init().

5.4.2.6 void IO::getTimeXML (char *inpfilename*[])

Definition at line 198 of file io.cc.

References DOCTYPE, MAXLINLEN, parseFloat(), and Run::time.

Referenced by getTime().

5.4.2.7 void IO::initxwd (int *i*)

Definition at line 15 of file io.cc.

References nxwdump.

5.4.2.8 double IO::parseFloat (xmlDocPtr *doc*, xmlNodePtr *cur*, char * *keyword*)

Definition at line 178 of file io.cc.

References Run::option.

Referenced by Domain::Domain().

5.4.2.9 int IO::parseFloat (xmlDocPtr *doc*, xmlNodePtr *cur*, char * *keyword*, REAL & *result*)

Definition at line 158 of file io.cc.

References Run::option, and REAL.

Referenced by getTimeXML().

5.4.2.10 int IO::parseInt (xmlDocPtr *doc*, xmlNodePtr *cur*, char * *keyword*)

Definition at line 138 of file io.cc.

References Run::option.

Referenced by getIntAttr().

5.4.2.11 int IO::parseWord (xmlDocPtr *doc*, xmlNodePtr *cur*, char * *keyword*, char * *result*)

Definition at line 117 of file io.cc.

References MAXLINLEN, and Run::option.

Referenced by Domain::Domain(), and Gui::readconf().

5.4.2.12 void IO::xwd (char * *windowname*)

Definition at line 19 of file io.cc.

References MAXLINLEN, nxwdump, and Run::option.

Referenced by Gui::dumpwindow().

5.4.3 Variable Documentation

5.4.3.1 int IO::ioutput

Definition at line 13 of file io.cc.

Referenced by Domain::run(), and Domain::save().

5.4.3.2 int IO::ioutput = 0

Definition at line 13 of file io.cc.

Referenced by Domain::run(), and Domain::save().

5.4.3.3 int IO::nxwdump = 0

Definition at line 14 of file io.cc.

Referenced by initxwd(), and xwd().

5.5 Palette Namespace Reference

Functions

- void **init** (REAL vmin, REAL vmax)
- void **pickcolor** (REAL var, REAL color[])

Variables

- REAL **vmn**
- REAL **vmx**
- REAL **vav**
- REAL **dvi**

5.5.1 Function Documentation

5.5.1.1 void Palette::init (REAL *vmin*, REAL *vmax*)

Definition at line 2189 of file gui.cc.

References dvi, vav, vmn, and vmx.

Referenced by Gui::display().

5.5.1.2 void Palette::pickcolor (REAL *var*, REAL *color*[])

Definition at line 2197 of file gui.cc.

References dvi, REAL, vav, vmn, and vmx.

Referenced by Gui::display().

5.5.2 Variable Documentation

5.5.2.1 REAL Palette::dvi

Definition at line 2185 of file gui.cc.

Referenced by init(), and pickcolor().

5.5.2.2 REAL Palette::vav

Definition at line 2185 of file gui.cc.

Referenced by init(), pickcolor(), Gui::showParticle(), and Gui::showSphere().

5.5.2.3 REAL Palette::vmn

Definition at line 2185 of file gui.cc.

Referenced by init(), and pickcolor().

5.5.2.4 REAL Palette::vmx

Definition at line 2185 of file gui.cc.

Referenced by init(), and pickcolor().

5.6 Potential Namespace Reference

Interaction-potential functions, such as LJ.

Functions

- REAL **invdist** (REAL x)
- void **strength** (REAL strength)
- REAL **strength** ()
- void **lengthscale** (REAL lengthscale)
- REAL **lengthscale** ()
- void **Cutoff** (REAL newcutoff)
- REAL **Cutoff** ()
- REAL **value** (REAL r)
- REAL **force** (REAL r)
- REAL **derivative** (REAL r)

Variables

- REAL **sigma** = 1.0
- REAL **eta** = 1.0
- REAL **cutoff** = 2.0
- const REAL **small** = 1.0e-20
- const REAL **large** = 1.0e20
- REAL **cutoff**
- REAL **sigma**
- REAL **eta**

5.6.1 Detailed Description

Interaction-potential functions, such as LJ.

5.6.2 Function Documentation

5.6.2.1 REAL Potential::Cutoff () [inline]

Definition at line 144 of file model.h.

References cutoff.

5.6.2.2 void Potential::Cutoff (REAL *newcutoff*) [inline]

Definition at line 143 of file model.h.

References cutoff.

5.6.2.3 REAL Potential::derivative (REAL *r*)

5.6.2.4 REAL Potential::force (REAL *r*) [inline]

Definition at line 156 of file model.h.

References eta, and sigma.

5.6.2.5 REAL Potential::invdist (REAL *x*)

Definition at line 78 of file model.cc.

5.6.2.6 REAL Potential::lengthscale () [inline]

Definition at line 142 of file model.h.

References sigma.

Referenced by Domain::Domain().

5.6.2.7 void Potential::lengthscale (REAL *lengthscale*) [inline]

Definition at line 139 of file model.h.

References sigma.

Referenced by Gui::display(), and Domain::Domain().

5.6.2.8 REAL Potential::strength () [inline]

Definition at line 138 of file model.h.

References eta.

Referenced by Domain::Domain().

5.6.2.9 void Potential::strength (REAL *strength*) [inline]

Definition at line 135 of file model.h.

References eta.

Referenced by Domain::Domain().

5.6.2.10 REAL Potential::value (REAL *r*) [inline]

Definition at line 145 of file model.h.

References cutoff, eta, large, REAL, sigma, and small.

Referenced by IO::getIntAttr().

5.6.3 Variable Documentation

5.6.3.1 REAL Potential::cutoff

Definition at line 61 of file model.cc.

Referenced by Cutoff(), and value().

5.6.3.2 REAL Potential::cutoff = 2.0

Definition at line 61 of file model.cc.

Referenced by Cutoff(), and value().

5.6.3.3 REAL Potential::eta

Definition at line 61 of file model.cc.

Referenced by force(), strength(), and value().

5.6.3.4 REAL Potential::eta = 1.0

Definition at line 61 of file model.cc.

Referenced by force(), strength(), and value().

5.6.3.5 const REAL Potential::large = 1.0e20

Definition at line 132 of file model.h.

Referenced by value().

5.6.3.6 REAL Potential::sigma

Definition at line 61 of file model.cc.

Referenced by force(), lengthscale(), and value().

5.6.3.7 REAL Potential::sigma = 1.0

Definition at line 61 of file model.cc.

Referenced by force(), lengthscale(), and value().

5.6.3.8 const REAL Potential::small = 1.0e-20

Definition at line 132 of file model.h.

Referenced by value().

5.7 Run Namespace Reference

Processing command-line and execution control parameters.

Classes

- struct **Time**

Functions

- void **testrnd** ()
- REAL **rnd** ()
- void **init** (int argc, char *argv[])
- void **readcmdline** (int argc, char *argv[])

Variables

- **Option option** = {0,0,0,0}
- void(*) **usage** ()
- char **programname** [MAXLINLEN]
- char **configfile** [MAXLINLEN]
- char **outputfile** [MAXLINLEN]
- char **outputname** [MAXLINLEN]
- char **inputfile** [MAXLINLEN]
- **Time time**
- **Option option**
- **Time time**
- void(*) **usage** ()

5.7.1 Detailed Description

Processing command-line and execution control parameters.

5.7.2 Function Documentation

5.7.2.1 void Run::init (int *argc*, char * *argv*[])

Definition at line 34 of file run.cc.

References configfile, Option::debug, IO::getTime(), Option::mesh, option, readcmdline(), Option::verbose, and Option::xterm.

Referenced by main().

5.7.2.2 void Run::readcmdline (int *argc*, char * *argv*[])

Definition at line 47 of file run.cc.

References configfile, Option::debug, i, inputfile, Option::mesh, option, outputfile, outputname, programname, and Option::verbose.

Referenced by init().

5.7.2.3 REAL Run::rnd ()

Definition at line 31 of file run.cc.

References REAL.

Referenced by testrnd().

5.7.2.4 void Run::testrnd ()

Definition at line 20 of file run.cc.

References i, REAL, and rnd().

5.7.3 Variable Documentation

5.7.3.1 char Run::configfile

Definition at line 14 of file run.cc.

Referenced by Domain::Domain(), init(), main(), readcmdline(), and Gui::readconf().

5.7.3.2 char Run::inputfile

Definition at line 14 of file run.cc.

Referenced by main(), and readcmdline().

5.7.3.3 struct Option Run::option

Definition at line 11 of file run.cc.

Referenced by Gui::animate(), Gui::display(), Domain::Domain(), IO::getCharAttr(), IO::getIntAttr(), IO::getIter(), IO::getTime(), GridContainer::init(), Gui::init(), init(), Gui::initdisp(), Gui::keyboard(), main(), Gui::Materials(), IO::parseFloat(), IO::parseInt(), IO::parseWord(), Pool< Element >::Pool(), readcmdline(), Gui::readconf(), Domain::run(), Domain::save(), IO::xwd(), and Domain::~Domain().

5.7.3.4 struct Option Run::option = {0,0,0,0}

Definition at line 11 of file run.cc.

Referenced by Gui::animate(), Gui::display(), Domain::Domain(), IO::getCharAttr(), IO::getIntAttr(), IO::getIter(), IO::getTime(), init(), Gui::init(), GridContainer::init(), Gui::initdisp(), Gui::keyboard(), main(), Gui::Materials(), IO::parseFloat(), IO::parseInt(), IO::parseWord(), Pool< Element >::Pool(), readcmdline(), Gui::readconf(), Domain::run(), Domain::save(), IO::xwd(), and Domain::~Domain().

5.7.3.5 char Run::outputfile

Definition at line 14 of file run.cc.

Referenced by readcmdline().

5.7.3.6 char Run::outputname

Definition at line 14 of file run.cc.

Referenced by Gui::helpCommand(), Gui::helpDisplay(), Gui::keyboard(), readcmdline(), and Domain::run().

5.7.3.7 char Run::programname

Definition at line 14 of file run.cc.

Referenced by readcmdline(), and usage().

5.7.3.8 struct Time Run::time

Definition at line 19 of file run.cc.

Referenced by Gui::animate(), Domain::boundary(), Gui::display(), Domain::Domain(), IO::getTime(), IO::getTimeXML(), Domain::injection(), Gui::keyboard(), Molecule::Move(), Domain::run(), Domain::save(), and Gui::toggleWindowDump().

5.7.3.9 struct Time Run::time

Definition at line 19 of file run.cc.

Referenced by Gui::animate(), Domain::boundary(), Gui::display(), Domain::Domain(), IO::getTime(), IO::getTimeXML(), Domain::injection(), Gui::keyboard(), Molecule::Move(), Domain::run(), Domain::save(), and Gui::toggleWindowDump().

5.7.3.10 void(*) Run::usage()

Definition at line 12 of file run.cc.

5.7.3.11 void(*) Run::usage()

Definition at line 12 of file run.cc.

5.8 Species Namespace Reference

Description of **Specie** (p. 108) and **Reaction** (p. 98) classes.

Classes

- class **Specie**
- struct **Gas**

  Gas *(p. 77) is a specie with some additional parameters.*

- struct **Reaction**

  Reaction *(p. 98) determines the two products only.*

Enumerations

- enum **Interaction** { **missed** = 0, **collided**, **reacted**, **annihalated** }

Variables

- int **nspecies**
- **Specie** ∗ **species**
- **Reaction** ∗ **reactions**
- int **nspecies**
- **Specie** ∗ **species**
- **Reaction** ∗ **reactions**

5.8.1 Detailed Description

Description of **Specie** (p. 108) and **Reaction** (p. 98) classes.

5.8.2 Enumeration Type Documentation

5.8.2.1 enum Species::Interaction

Enumerator:

missed

collided

reacted two reactants -> two products

annihalated two reactants -> one product

Definition at line 164 of file species.h.

5.8.3 Variable Documentation

5.8.3.1 int Species::nspecies

Definition at line 6 of file species.cc.

Referenced by Domain::Domain(), and Species::Reaction::Outcome::Outcome().

5.8.3.2 int Species::nspecies

Definition at line 6 of file species.cc.

Referenced by Domain::Domain(), and Species::Reaction::Outcome::Outcome().

5.8.3.3 Reaction* Species::reactions

Definition at line 8 of file species.cc.

Referenced by Domain::Domain(), and Domain::interact().

5.8.3.4 Reaction* Species::reactions

Definition at line 8 of file species.cc.

Referenced by Domain::Domain(), and Domain::interact().

5.8.3.5 Specie* Species::species

Definition at line 7 of file species.cc.

Referenced by Domain::boundary(), Gui::display(), Domain::Domain(), Domain::injection(), Domain::interact(), Molecule::KineticEnergy(), Molecule::Move(), Domain::run(), and Domain::save().

5.8.3.6 Specie* Species::species

Definition at line 7 of file species.cc.

Referenced by Domain::boundary(), Gui::display(), Domain::Domain(), Domain::injection(), Domain::interact(), Molecule::KineticEnergy(), Molecule::Move(), Domain::run(), and Domain::save().

5.9 std Namespace Reference

Chapter 6

ReMoDy Class Documentation

6.1 Boundary Class Reference

Domain (p. 70) **Boundary** (p. 51) Definition.

`#include <domain.h>`

Public Member Functions

- **Boundary** ()
- ~**Boundary** ()
- REAL **Area** ()
- void **Area** (REAL a)
- REAL **Temperature** ()
- void **Temperature** (REAL t)
- void **init** (int **i**, int j, REAL ymin[ ], REAL ymax[ ], int nspecies)
- void **Inject** (int specietype, REAL velocity, **Collection**< **Molecule** > ∗molecules)

  Injecting a specie from accross the boundary with a specified velocity.

Public Attributes

- int **idir**
- int **iside**

  direction: 0,1,2; side: 0,1

- REAL ∗ **xmin**
- REAL ∗ **xmax**

  min/max bounds

- **BoundaryTypes type**
- bool **adiabatic**

  flag for adiabatic boundary

- REAL **area**

- REAL **temperature**
- **List**< Gas > **gases**

  list of gases across the boundary

- Reaction * **reactions**

  boundary reactions with all species

6.1.1 Detailed Description

Domain (p. 70) **Boundary** (p. 51) Definition.

Definition at line 16 of file domain.h.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 Boundary::Boundary () [inline]

Definition at line 26 of file domain.h.

References adiabatic, area, elasticBoundary, idir, iside, reactions, temperature, type, xmax, and xmin.

6.1.2.2 Boundary::~Boundary () [inline]

Definition at line 35 of file domain.h.

6.1.3 Member Function Documentation

6.1.3.1 REAL Boundary::Area () [inline]

Definition at line 38 of file domain.h.

References area.

6.1.3.2 void Boundary::Area (REAL *a*) [inline]

Definition at line 39 of file domain.h.

References area.

6.1.3.3 REAL Boundary::Temperature () [inline]

Definition at line 40 of file domain.h.

References temperature.

6.1.3.4 void Boundary::Temperature (REAL *t*) [inline]

Definition at line 41 of file domain.h.

References temperature.

6.1.3.5 void Boundary::init (int *i*, int *j*, REAL *ymin*[ ], REAL *ymax*[ ], int *nspecies*)

Definition at line 23 of file domain.cc.

References idir, iside, reactions, xmax, and xmin.

6.1.3.6 void Boundary::Inject (int *specietype*, REAL *velocity*, Collection< Molecule > * *molecules*)

Injecting a specie from accross the boundary with a specified velocity.

Definition at line 29 of file domain.cc.

References Collection< Element >::append(), Molecule::Coordinate(), DIM, i, idir, iside, GridContainer::put(), REAL, RND, Molecule::Type(), Molecule::Velocity(), xmax, and xmin.

6.1.4 Member Data Documentation

6.1.4.1 int Boundary::idir

Definition at line 17 of file domain.h.

Referenced by Boundary(), init(), and Inject().

6.1.4.2 int Boundary::iside

direction: 0,1,2; side: 0,1

Definition at line 17 of file domain.h.

Referenced by Boundary(), init(), and Inject().

6.1.4.3 REAL* Boundary::xmin

Definition at line 18 of file domain.h.

Referenced by Boundary(), init(), and Inject().

6.1.4.4 REAL * Boundary::xmax

min/max bounds

Definition at line 18 of file domain.h.

Referenced by Boundary(), init(), and Inject().

6.1.4.5 BoundaryTypes Boundary::type

Definition at line 19 of file domain.h.

Referenced by Boundary(), and Domain::BoundaryType().

6.1.4.6 bool Boundary::adiabatic

flag for adiabatic boundary

Definition at line 20 of file domain.h.

Referenced by Boundary().

6.1.4.7 REAL Boundary::area

Definition at line 22 of file domain.h.

Referenced by Area(), and Boundary().

6.1.4.8 REAL Boundary::temperature

Definition at line 22 of file domain.h.

Referenced by Boundary(), and Temperature().

6.1.4.9 List<Gas> Boundary::gases

list of gases across the boundary

Definition at line 24 of file domain.h.

6.1.4.10 Reaction* Boundary::reactions

boundary reactions with all species

Definition at line 25 of file domain.h.

Referenced by Boundary(), and init().

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

- **domain.h**
- **domain.cc**

6.2 Collection< Element > Class Template Reference

`#include <collection.h>`

Public Member Functions

- **Collection** ()
- ~**Collection** ()
- int **number** ()
- int **maxnumber** ()
- void **setFirst** ()
- void **setFirstLast** ()
- void **goFirst** ()
- void **goFirstNext** ()
- void **FirstNext** ()
- void **FirstPrev** ()
- bool **isFirstLast** ()
- void **goLastNext** ()
- void **goLast** ()
- void **goNext** ()
- void **goPrev** ()
- bool **isFirst** ()
- bool **isLast** ()
- void **LastNext** ()
- void **LastPrev** ()
- void **go** (**Item**< Element > ∗p)
- Element ∗ **First** ()
- Element ∗ **Last** ()
- Element ∗ **Next** ()
- Element ∗ **getNext** ()
- Element ∗ **Prev** ()
- Element ∗ **getPrev** ()
- Element ∗ **Current** ()
- **Item**< Element > ∗ **getItem** ()
- **Item**< Element > ∗ **getFirstItem** ()
- **Item**< Element > ∗ **getLastItem** ()
- bool **init** (int **n**)
- Element ∗ **insert** ()
- bool **insert** (Element ∗element)
- Element ∗ **append** ()
- bool **append** (Element ∗element)
- bool **remove** ()

Private Attributes

- int **mitems**
- int **nitems**
- **Item**< Element > ∗ **items**
- **Item**< Element > ∗ **dead**
- **Item**< Element > ∗ **first**
- **Item**< Element > ∗ **last**
- **Item**< Element > ∗ **current**

6.2.1 Detailed Description

template<class Element> class Collection< Element >

Definition at line 24 of file collection.h.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 template<class Element> Collection< Element >::Collection () [inline]

Definition at line 69 of file collection.h.

References Collection< Element >::current, Collection< Element >::dead, Collection< Element >::first, Collection< Element >::last, Collection< Element >::mitems, and Collection< Element >::nitems.

6.2.2.2 template<class Element> Collection< Element >::~Collection () [inline]

Definition at line 74 of file collection.h.

References Collection< Element >::current, Collection< Element >::dead, Collection< Element >::first, Collection< Element >::last, Collection< Element >::mitems, and Collection< Element >::nitems.

6.2.3 Member Function Documentation

6.2.3.1 template<class Element> int Collection< Element >::number () [inline]

Definition at line 33 of file collection.h.

Referenced by Gui::display(), Gui::initdisp(), Domain::run(), and Domain::save().

6.2.3.2 template<class Element> int Collection< Element >::maxnumber () [inline]

Definition at line 34 of file collection.h.

Referenced by Domain::run().

6.2.3.3 template<class Element> void Collection< Element >::setFirst () [inline]

Definition at line 35 of file collection.h.

6.2.3.4 template<class Element> void Collection< Element >::setFirstLast () [inline]

Definition at line 36 of file collection.h.

6.2.3.5 template<class Element> void Collection< Element >::goFirst () [inline]

Definition at line 37 of file collection.h.

Referenced by Domain::computeBounds(), Domain::run(), and Domain::save().

6.2.3.6 template<class Element> void Collection< Element >::goFirstNext () [inline]

Definition at line 38 of file collection.h.

6.2.3.7 template<class Element> void Collection< Element >::FirstNext () [inline]

Definition at line 39 of file collection.h.

6.2.3.8 template<class Element> void Collection< Element >::FirstPrev () [inline]

Definition at line 40 of file collection.h.

6.2.3.9 template<class Element> bool Collection< Element >::isFirstLast () [inline]

Definition at line 41 of file collection.h.

6.2.3.10 template<class Element> void Collection< Element >::goLastNext () [inline]

Definition at line 42 of file collection.h.

6.2.3.11 template<class Element> void Collection< Element >::goLast () [inline]

Definition at line 43 of file collection.h.

6.2.3.12 template<class Element> void Collection< Element >::goNext () [inline]

Definition at line 44 of file collection.h.

Referenced by Domain::computeBounds(), Domain::run(), and Domain::save().

6.2.3.13 template<class Element> void Collection< Element >::goPrev () [inline]

Definition at line 45 of file collection.h.

6.2.3.14 template<class Element> bool Collection< Element >::isFirst () [inline]

Definition at line 46 of file collection.h.

Referenced by Domain::computeBounds(), Domain::run(), and Domain::save().

6.2.3.15 template<class Element> bool Collection< Element >::isLast () [inline]

Definition at line 47 of file collection.h.

6.2.3.16 template<class Element> void Collection< Element >::LastNext () [inline]

Definition at line 48 of file collection.h.

6.2.3.17 template<class Element> void Collection< Element >::LastPrev () [inline]

Definition at line 49 of file collection.h.

6.2.3.18 template<class Element> void Collection< Element >::go (Item< Element > * *p*) [inline]

Definition at line 50 of file collection.h.

6.2.3.19 template<class Element> Element* Collection< Element >::First () [inline]

Definition at line 51 of file collection.h.

6.2.3.20 template<class Element> Element* Collection< Element >::Last () [inline]

Definition at line 52 of file collection.h.

6.2.3.21 template<class Element> Element* Collection< Element >::Next () [inline]

Definition at line 53 of file collection.h.

6.2.3.22 template<class Element> Element* Collection< Element >::getNext () [inline]

Definition at line 54 of file collection.h.

6.2.3.23 template<class Element> Element* Collection< Element >::Prev () [inline]

Definition at line 55 of file collection.h.

6.2.3.24 template<class Element> Element* Collection< Element >::getPrev () [inline]

Definition at line 56 of file collection.h.

6.2.3.25 template<class Element> Element* Collection< Element >::Current () [inline]

Definition at line 57 of file collection.h.

Referenced by Domain::computeBounds(), Domain::run(), and Domain::save().

6.2.3.26 template<class Element> Item<Element>* Collection< Element >::getItem () [inline]

Definition at line 58 of file collection.h.

6.2.3.27 template<class Element> Item<Element>* Collection< Element >::getFirstItem () [inline]

Definition at line 59 of file collection.h.

6.2.3.28 template<class Element> Item<Element>* Collection< Element >::getLastItem () [inline]

Definition at line 60 of file collection.h.

6.2.3.29 template<class Element> bool Collection< Element >::init (int *n*) [inline]

Definition at line 80 of file collection.h.

References Collection< Element >::current, Collection< Element >::dead, Collection< Element >::first, i, Collection< Element >::last, Collection< Element >::mitems, and Collection< Element >::nitems.

Referenced by Domain::Domain().

6.2.3.30 template<class Element> Element * Collection< Element >::insert () [inline]

Definition at line 99 of file collection.h.

References Collection< Element >::current, Collection< Element >::dead, Item< Element >::element, Collection< Element >::first, Collection< Element >::last, Item< Element >::next, Collection< Element >::nitems, and Item< Element >::prev.

Referenced by Domain::boundary(), and Collection< Element >::insert().

6.2.3.31 template<class Element> bool Collection< Element >::insert (Element * *element*) [inline]

Definition at line 131 of file collection.h.

References Collection< Element >::insert().

6.2.3.32 template<class Element> Element * Collection< Element >::append () [inline]

Definition at line 138 of file collection.h.

References Collection< Element >::current, Collection< Element >::dead, Collection< Element >::first, Collection< Element >::last, Item< Element >::next, Collection< Element >::nitems, and Item< Element >::prev.

Referenced by Collection< Element >::append(), and Boundary::Inject().

6.2.3.33 template<class Element> bool Collection< Element >::append (Element * *element*) [inline]

Definition at line 170 of file collection.h.

References Collection< Element >::append().

6.2.3.34 template<class Element> bool Collection< Element >::remove () [inline]

Definition at line 177 of file collection.h.

References Collection< Element >::current, Collection< Element >::dead, Collection< Element >::first, Collection< Element >::last, Item< Element >::next, and Collection< Element >::nitems.

Referenced by Domain::run().

6.2.4 Member Data Documentation

6.2.4.1 template<class Element> int Collection< Element >::mitems [private]

Definition at line 25 of file collection.h.

Referenced by Collection< Element >::Collection(), Collection< Element >::init(), Collection< Molecule >::maxnumber(), and Collection< Element >::~Collection().

6.2.4.2 template<class Element> int Collection< Element >::nitems [private]

Definition at line 25 of file collection.h.

Referenced by Collection< Element >::append(), Collection< Element >::Collection(), Collection< Element >::init(), Collection< Element >::insert(), Collection< Molecule >::number(), Collection< Element >::remove(), and Collection< Element >::~Collection().

6.2.4.3 template<class Element> struct Item< Element >* Collection< Element >::items [read, private]

Definition at line 26 of file collection.h.

6.2.4.4 template<class Element> struct Item< Element > * Collection< Element >::dead [read, private]

Definition at line 26 of file collection.h.

Referenced by Collection< Element >::append(), Collection< Element >::Collection(), Collection< Element >::init(), Collection< Element >::insert(), Collection< Element >::remove(), and Collection< Element >::~Collection().

6.2.4.5 template<class Element> struct Item< Element > * Collection< Element >::first [read, private]

Definition at line 26 of file collection.h.

Referenced by Collection< Element >::append(), Collection< Element >::Collection(), Collection< Molecule >::First(), Collection< Molecule >::FirstNext(), Collection< Molecule >::FirstPrev(), Collection< Molecule >::getFirstItem(), Collection< Molecule >::goFirst(), Collection< Molecule >::goFirstNext(), Collection< Element >::init(), Collection< Element >::insert(), Collection< Molecule >::isFirst(), Collection< Molecule >::isFirstLast(), Collection< Element >::remove(), Collection< Molecule >::setFirst(), Collection< Molecule >::setFirstLast(), and Collection< Element >::~Collection().

6.2.4.6 template<class Element> struct Item< Element > * Collection< Element >::last [read, private]

Definition at line 26 of file collection.h.

Referenced by Collection< Element >::append(), Collection< Element >::Collection(), Collection< Molecule >::getLastItem(), Collection< Molecule >::goLast(), Collection< Molecule >::goLastNext(), Collection< Element >::init(), Collection< Element >::insert(), Collection< Molecule >::isFirstLast(), Collection< Molecule >::isLast(), Collection< Molecule >::Last(), Collection< Molecule >::LastNext(), Collection< Molecule >::LastPrev(), Collection< Element >::remove(), Collection< Molecule >::setFirst(), Collection< Molecule >::setFirstLast(), and Collection< Element >::~Collection().

6.2.4.7 template<class Element> struct Item< Element > * Collection< Element >::current [read, private]

Definition at line 26 of file collection.h.

Referenced by Collection< Element >::append(), Collection< Element >::Collection(), Collection< Molecule >::Current(), Collection< Molecule >::getItem(), Collection< Molecule >::getNext(), Collection< Molecule >::getPrev(), Collection< Molecule >::go(), Collection< Molecule >::goFirst(), Collection< Molecule >::goFirstNext(), Collection< Molecule >::goLast(), Collection< Molecule >::goLastNext(), Collection< Molecule >::goNext(), Collection< Molecule >::goPrev(), Collection< Element >::init(), Collection< Element >::insert(), Collection< Molecule >::isFirst(), Collection< Molecule >::isLast(), Collection< Molecule >::Next(), Collection< Molecule >::Prev(), Collection< Element >::remove(), and Collection< Element >::~Collection().

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

- **collection.h**

6.3 Gui::ColorScale Struct Reference

`#include <gui.h>`

Public Attributes

- int **relative**:1

6.3.1 Detailed Description

Definition at line 104 of file gui.h.

6.3.2 Member Data Documentation

6.3.2.1 int Gui::ColorScale::relative

Definition at line 105 of file gui.h.

Referenced by Gui::initdisp().

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

- **gui.h**

6.4 Container< Element > Class Template Reference

`#include <collection.h>`

Public Member Functions

- **Container** ()
- ~**Container** ()
- int **number** ()
- void **setFirst** ()
- void **setFirstLast** ()
- void **goFirst** ()
- void **goFirstNext** ()
- void **FirstNext** ()
- void **FirstPrev** ()
- bool **isFirstLast** ()
- void **goLastNext** ()
- void **goLast** ()
- void **goNext** ()
- void **goPrev** ()
- bool **isFirst** ()
- bool **isLast** ()
- void **LastNext** ()
- void **LastPrev** ()
- void **go** (**Ptr**< Element > ∗p)
- Element ∗ **First** ()
- Element ∗ **Last** ()
- Element ∗ **Next** ()
- Element ∗ **getNext** ()
- Element ∗ **Prev** ()
- Element ∗ **getPrev** ()
- Element ∗ **Current** ()
- **Ptr**< Element > ∗ **getPtr** ()
- **Ptr**< Element > ∗ **getFirstPtr** ()
- **Ptr**< Element > ∗ **getLastPtr** ()
- bool **insert** (Element ∗element)
- bool **insert** ()
- bool **append** (Element ∗element)
- bool **append** ()
- bool **link** (**Ptr**< Element > ∗ptr)
- bool **unlink** (**Ptr**< Element > ∗ptr)
- bool **remove** ()
- bool **remove** (**Ptr**< Element > ∗ptr)
- bool **checkPool** (char ∗msg, **Pool**< Element > ∗pool)

Private Attributes

- int **nptrs**
- **Ptr**< Element > ∗ **first**
- **Ptr**< Element > ∗ **last**
- **Ptr**< Element > ∗ **current**

6.4.1 Detailed Description

template<class Element> class Container< Element >

Definition at line 311 of file collection.h.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 template<class Element> Container< Element >::Container () [inline]

Definition at line 2 of file container.h.

References Container< Element >::current, Container< Element >::first, Container< Element >::last, and Container< Element >::nptrs.

6.4.2.2 template<class Element> Container< Element >::~Container () [inline]

Definition at line 7 of file container.h.

References Container< Element >::current, Container< Element >::first, Container< Element >::last, and Container< Element >::nptrs.

6.4.3 Member Function Documentation

6.4.3.1 template<class Element> int Container< Element >::number () [inline]

Definition at line 321 of file collection.h.

References Pool< Element >::nptrs.

Referenced by Domain::interaction().

6.4.3.2 template<class Element> void Container< Element >::setFirst () [inline]

Definition at line 322 of file collection.h.

Referenced by Domain::interaction().

6.4.3.3 template<class Element> void Container< Element >::setFirstLast () [inline]

Definition at line 323 of file collection.h.

Referenced by Domain::interaction().

6.4.3.4 template<class Element> void Container< Element >::goFirst () [inline]

Definition at line 324 of file collection.h.

Referenced by Domain::interaction().

6.4.3.5 template<class Element> void Container< Element >::goFirstNext () [inline]

Definition at line 325 of file collection.h.

6.4.3.6 template<class Element> void Container< Element >::FirstNext () [inline]

Definition at line 326 of file collection.h.

Referenced by Domain::interaction().

6.4.3.7 template<class Element> void Container< Element >::FirstPrev () [inline]

Definition at line 327 of file collection.h.

6.4.3.8 template<class Element> bool Container< Element >::isFirstLast () [inline]

Definition at line 328 of file collection.h.

Referenced by Domain::interaction().

6.4.3.9 template<class Element> void Container< Element >::goLastNext () [inline]

Definition at line 329 of file collection.h.

6.4.3.10 template<class Element> void Container< Element >::goLast () [inline]

Definition at line 330 of file collection.h.

6.4.3.11 template<class Element> void Container< Element >::goNext () [inline]

Definition at line 331 of file collection.h.

Referenced by Domain::interaction().

6.4.3.12 template<class Element> void Container< Element >::goPrev () [inline]

Definition at line 332 of file collection.h.

6.4.3.13 template<class Element> bool Container< Element >::isFirst () [inline]

Definition at line 333 of file collection.h.

Referenced by Domain::interaction().

6.4.3.14 template<class Element> bool Container< Element >::isLast () [inline]

Definition at line 334 of file collection.h.

Referenced by Domain::interaction().

6.4.3.15 template<class Element> void Container< Element >::LastNext () [inline]

Definition at line 335 of file collection.h.

6.4.3.16 template<class Element> void Container< Element >::LastPrev () [inline]

Definition at line 336 of file collection.h.

6.4.3.17 template<class Element> void Container< Element >::go (Ptr< Element > * *p*) [inline]

Definition at line 337 of file collection.h.

6.4.3.18 template<class Element> Element* Container< Element >::First () [inline]

Definition at line 338 of file collection.h.

References Ptr< Element >::element.

Referenced by Domain::interaction().

6.4.3.19 template<class Element> Element* Container< Element >::Last () [inline]

Definition at line 339 of file collection.h.

6.4.3.20 template<class Element> Element* Container< Element >::Next () [inline]

Definition at line 340 of file collection.h.

6.4.3.21 template<class Element> Element* Container< Element >::getNext () [inline]

Definition at line 341 of file collection.h.

6.4.3.22 template<class Element> Element* Container< Element >::Prev () [inline]

Definition at line 342 of file collection.h.

6.4.3.23 template<class Element> Element* Container< Element >::getPrev () [inline]

Definition at line 343 of file collection.h.

6.4.3.24 template<class Element> Element* Container< Element >::Current () [inline]

Definition at line 344 of file collection.h.

Referenced by Domain::interaction().

6.4.3.25 template<class Element> Ptr<Element>* Container< Element >::getPtr () [inline]

Definition at line 345 of file collection.h.

6.4.3.26 template<class Element> Ptr<Element>* Container< Element >::getFirstPtr () [inline]

Definition at line 346 of file collection.h.

6.4.3.27 template<class Element> Ptr<Element>* Container< Element >::getLastPtr () [inline]

Definition at line 347 of file collection.h.

6.4.3.28 template<class Element> bool Container< Element >::insert (Element * *element*) [inline]

Definition at line 153 of file container.h.

References Container< Element >::current, and Container< Element >::insert().

6.4.3.29 template<class Element> bool Container< Element >::insert () [inline]

Definition at line 117 of file container.h.

References Container< Element >::current, Container< Element >::first, Pool< Element >::get(), Container< Element >::last, Ptr< Element >::next, Container< Element >::nptrs, GridContainer::pool, and Ptr< Element >::prev.

Referenced by Container< Element >::insert().

6.4.3.30 template<class Element> bool Container< Element >::append (Element * *element*) [inline]

Definition at line 56 of file container.h.

References Container< Element >::append(), and Container< Element >::last.

6.4.3.31 template<class Element> bool Container< Element >::append () [inline]

Definition at line 23 of file container.h.

References Container< Element >::current, Container< Element >::first, Pool< Element >::get(), Container< Element >::last, Container< Element >::nptrs, GridContainer::pool, and Ptr< Element >::prev.

Referenced by Container< Element >::append().

6.4.3.32 template<class Element> bool Container< Element >::link (Ptr< Element > * *ptr*) [inline]

Definition at line 63 of file container.h.

References Container< Element >::current, Container< Element >::first, Container< Element >::last, Ptr< Element >::next, Container< Element >::nptrs, and Ptr< Element >::prev.

6.4.3.33 template<class Element> bool Container< Element >::unlink (Ptr< Element > * *ptr*) [inline]

Definition at line 86 of file container.h.

References Container< Element >::current, Container< Element >::first, Container< Element >::last, Ptr< Element >::next, Container< Element >::nptrs, and Ptr< Element >::prev.

6.4.3.34 template<class Element> bool Container< Element >::remove () [inline]

Definition at line 162 of file container.h.

References Container< Element >::current, Container< Element >::first, Container< Element >::last, Container< Element >::nptrs, GridContainer::pool, and Pool< Element >::put().

Referenced by Domain::boundary(), and Domain::interaction().

6.4.3.35 template<class Element> bool Container< Element >::remove (Ptr< Element > * *ptr*) [inline]

Definition at line 183 of file container.h.

References Container< Element >::current, Container< Element >::first, Container< Element >::last, Container< Element >::nptrs, GridContainer::pool, and Pool< Element >::put().

6.4.3.36 template<class Element> bool Container< Element >::checkPool (char * *msg*, Pool< Element > * *pool*) [inline]

Definition at line 12 of file container.h.

References Pool< Element >::check(), Container< Element >::current, Container< Element >::first, Pool< Element >::hook, Ptr< Element >::next, Container< Element >::nptrs, and GridContainer::pool.

6.4.4 Member Data Documentation

6.4.4.1 template<class Element> int Container< Element >::nptrs [private]

Definition at line 312 of file collection.h.

Referenced by Container< Element >::append(), Container< Element >::checkPool(), Container< Element >::Container(), Container< Element >::insert(), Container< Element >::link(), Container< Element >::remove(), Container< Element >::unlink(), and Container< Element >::~Container().

6.4.4.2 template<class Element> Ptr<Element>* Container< Element >::first [private]

Definition at line 315 of file collection.h.

Referenced by Container< Element >::append(), Container< Element >::checkPool(), Container< Element >::Container(), Container< Element >::insert(), Container< Element >::link(), Container< Element >::remove(), Container< Element >::unlink(), and Container< Element >::~Container().

6.4.4.3 template<class Element> Ptr<Element> * Container< Element >::last [private]

Definition at line 315 of file collection.h.

Referenced by Container< Element >::append(), Container< Element >::Container(), Container< Element >::insert(), Container< Element >::link(), Container< Element >::remove(), Container< Element >::unlink(), and Container< Element >::~Container().

6.4.4.4 template<class Element> Ptr<Element> * Container< Element >::current [private]

Definition at line 315 of file collection.h.

Referenced by Container< Element >::append(), Container< Element >::checkPool(), Container< Element >::Container(), Container< Element >::insert(), Container< Element >::link(), Container< Element >::remove(), Container< Element >::unlink(), and Container< Element >::~Container().

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

- **collection.h**
- **container.h**

6.5 Domain Class Reference

Computational **Domain** (p. 70).

`#include <domain.h>`

Public Member Functions

- **Domain** (char *filename)

  Domain *(p. 70) constructor creates an instance of a domain.*

- ~**Domain** ()
- void **BoundaryType** (enum **BoundaryTypes** b, int idir, int inside)
- enum **BoundaryTypes BoundaryType** (int idir, int inside)
- void **setMinBound** (int **i**, REAL x)
- void **setMaxBound** (int **i**, REAL x)
- REAL **minBound** (int **i**)
- REAL **maxBound** (int **i**)
- int **computeBounds** (REAL x0[], REAL x1[])
- **Collection**< **Molecule** > * **Molecules** ()
- int **run** (int niter)

  The ***Domain*** *(p. 70) solver: running* **niter** *iterations.*

- enum **BoundaryTypes boundary** (**Molecule** *molecule)
- void **injection** ()
- void **interaction** ()
- **Interaction interact** (**Molecule** *a, **Molecule** *b)
- void **save** (char *taskname)

Private Attributes

- int * **distribution**

  molecule counts for all species

- REAL **xmin** [DIM]
- REAL **xmax** [DIM]

  domain bounds

- char **boundaryName** [maxBoundaryTypes][WORDLENGTH]

  to be depricated

- **Boundary boundaries** [DIM][2]

  six boundaries of a hexahedral domain

- **Collection**< **Molecule** > **molecules**

  Collection *(p. 55) of molecules.*

- **Pool**< **Molecule** > * **pool**

  Molecule *(p. 87) pointer pool.*

6.5.1 Detailed Description

Computational **Domain** (p. 70).

Implements main data types inside a computational domain

Definition at line 51 of file domain.h.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 Domain::Domain (char * *filename*)

Domain (p. 70) constructor creates an instance of a domain.

get domain specs from XML configfile

read XML input file

Loop over file records

set xterm output mode:

Read the number of molecules:

Read species:

load reactions

Read domain specs:

Read boundary specs

Read boundary reactions:

< probability for the formation of these products as opposed to different products If only one pair of products exists the probability = 1

load molecules from the gzipped data file:

Definition at line 56 of file domain.cc.

References Species::Reaction::Add(), Geom::area(), AtomicMassUnit, AvogadroNumber, BoltzmannConstant, boundaries, boundary(), boundaryName, GridContainer::cellsize, Run::configfile, Species::Gas::density, DIM, distribution, DOCTYPE, elasticBoundary, Species::Reaction::Outcome::Enthalpy(), Species::Reaction::First(), GasConstant, IO::getCharAttr(), i, Species::Specie::Id(), Collection< Element >::init(), insideBoundary, Potential::lengthscale(), Species::Specie::Mass(), maxBound(), maxBoundaryTypes, MAXLINLEN, minBound(), molecules, Species::nspecies, openBoundary, Run::option, IO::parseFloat(), IO::parseWord(), periodicBoundary, pool, Species::Reaction::Outcome::Probability(), Species::Reaction::Outcome::Product(), Species::reactions, REAL, setMaxBound(), setMinBound(), Species::Specie::Size(), Gui::size, Species::species, Potential::strength(), Run::time, TINY, VOIDSPECIE, WORDLENGTH, xmax, xmin, and Option::xterm.

6.5.2.2 Domain::~Domain ()

Definition at line 784 of file domain.cc.

References Run::option.

6.5.3 Member Function Documentation

6.5.3.1 void Domain::BoundaryType (enum BoundaryTypes *b*, int *idir*, int *inside*) [inline]

Definition at line 61 of file domain.h.

References boundaries, DIM, and Boundary::type.

6.5.3.2 enum BoundaryTypes Domain::BoundaryType (int *idir*, int *inside*) [inline]

Definition at line 68 of file domain.h.

References boundaries, DIM, and Boundary::type.

6.5.3.3 void Domain::setMinBound (int *i*, REAL *x*) [inline]

Definition at line 75 of file domain.h.

References xmin.

Referenced by Domain(), and Gui::initdisp().

6.5.3.4 void Domain::setMaxBound (int *i*, REAL *x*) [inline]

Definition at line 76 of file domain.h.

References xmax.

Referenced by Domain(), and Gui::initdisp().

6.5.3.5 REAL Domain::minBound (int *i*) [inline]

Definition at line 77 of file domain.h.

References xmin.

Referenced by Domain(), and Gui::readconf().

6.5.3.6 REAL Domain::maxBound (int *i*) [inline]

Definition at line 78 of file domain.h.

References xmax.

Referenced by Domain(), and Gui::readconf().

6.5.3.7 int Domain::computeBounds (REAL *x0*[ ], REAL *x1*[ ])

Definition at line 653 of file domain.cc.

References Molecule::Coordinates(), Collection< Element >::Current(), Collection< Element >::goFirst(), Collection< Element >::goNext(), i, Collection< Element >::isFirst(), LARGE, molecules, and REAL.

6.5.3.8 Collection<Molecule>* Domain::Molecules () [inline]

Definition at line 82 of file domain.h.

References molecules.

Referenced by Gui::display(), and Gui::initdisp().

6.5.3.9 int Domain::run (int *niter*)

The **Domain** (p. 70) solver: running **niter** iterations.

The main loop over niter iterations or till the end of time:

Nested looping over all molecules:

Definition at line 673 of file domain.cc.

References AtomicMassUnit, BoltzmannConstant, boundary(), Collection< Element >::Current(), distribution, ESC, Collection< Element >::goFirst(), Collection< Element >::goNext(), i, injection(), interaction(), Molecule::InternalEnergy(), IO::ioutput, Collection< Element >::isFirst(), Molecule::KineticEnergy(), LENGTH, Collection< Element >::maxnumber(), molecules, Molecule::Move(), Collection< Element >::number(), Run::option, Run::outputname, GridContainer::put(), REAL, Collection< Element >::remove(), save(), Species::Specie::Size(), SMALL, Species::species, Gui::step, Run::time, Molecule::Type(), and VOIDSPECIE.

Referenced by Gui::animate(), Gui::keyboard(), main(), and Gui::runmany().

6.5.3.10 enum BoundaryTypes Domain::boundary (Molecule * *molecule*)

< do reaction: A + B =

< new molecule A type

< new molecule B type

Definition at line 788 of file domain.cc.

References boundaries, Molecule::Coordinates(), Species::Specie::Cp(), Gui::dx, elasticBoundary, Species::Reaction::Outcome::Enthalpy(), Species::Reaction::First(), GasConstant, i, Collection< Element >::insert(), insideBoundary, Molecule::InternalEnergy(), Molecule::KineticEnergy(), LENGTH, Species::Specie::Mass(), molecules, MULC, Species::Reaction::Next(), GridContainer::nodes, openBoundary, periodicBoundary, Species::Reaction::Outcome::Probability(), Species::Reaction::Outcome::Product(), Species::reacted, REAL, Container< Element >::remove(), RND, Species::Specie::Size(), SMALL, Species::species, Species::Reaction::Outcome::Time(), Run::time, Molecule::Type(), Molecule::Velocity(), VOIDSPECIE, xmax, xmin, and y.

Referenced by Domain(), injection(), and run().

6.5.3.11 void Domain::injection ()

Definition at line 958 of file domain.cc.

References Geom::area(), AtomicMassUnit, BoltzmannConstant, boundaries, boundary(), Species::Gas::density, Species::Specie::Mass(), molecules, PI, REAL, RND, Species::Gas::specie, Species::species, Gui::step, and Run::time.

Referenced by run().

6.5.3.12 void Domain::interaction ()

Definition at line 1000 of file domain.cc.

References Species::annihalated, Container< Element >::Current(), Gui::dx, Container< Element >::First(), Container< Element >::FirstNext(), Container< Element >::goFirst(), Container< Element >::goNext(), i, GridContainer::index(), interact(), Container< Element >::isFirst(), Container< Element >::isFirstLast(), Container< Element >::isLast(), Species::missed, GridContainer::ncells, Gui::nodes, Container< Element >::number(), REAL, Container< Element >::remove(), Container< Element >::setFirst(), Container< Element >::setFirstLast(), Molecule::Type(), VOIDSPECIE, xmax, and xmin.

Referenced by run().

6.5.3.13 Interaction Domain::interact (Molecule * *a*, Molecule * *b*)

DDD

Definition at line 1078 of file domain.cc.

References Species::Reaction::Outcome::ActivationEnergy(), Species::annihalated, Species::collided, Molecule::Coordinates(), Species::Specie::Cp(), Species::Reaction::Outcome::Enthalpy(), Species::Reaction::First(), GasConstant, i, Species::Specie::Id(), Molecule::InternalEnergy(), Species::Specie::Mass(), Species::missed, Species::Reaction::Next(), Species::Reaction::Outcome::Probability(), Species::Reaction::Outcome::Product(), Species::reacted, Species::reactions, REAL, RND, SCLP, Species::Specie::Size(), SMALL, Species::species, Molecule::Type(), Molecule::Velocity(), and VOIDSPECIE.

Referenced by interaction().

6.5.3.14 void Domain::save (char * *taskname*)

Definition at line 1378 of file domain.cc.

References Molecule::Coordinate(), Collection< Element >::Current(), Collection< Element >::goFirst(), Collection< Element >::goNext(), Species::Specie::Id(), IO::ioutput, Collection< Element >::isFirst(), MAXLINLEN, molecules, Collection< Element >::number(), Run::option, Species::Specie::Size(), Species::species, Run::time, Molecule::Type(), Molecule::Velocity(), and VOIDSPECIE.

Referenced by Gui::keyboard(), main(), and run().

6.5.4 Member Data Documentation

6.5.4.1 int* Domain::distribution [private]

molecule counts for all species

Definition at line 52 of file domain.h.

Referenced by Domain(), and run().

6.5.4.2 REAL Domain::xmin[DIM] [private]

Definition at line 53 of file domain.h.

Referenced by boundary(), Domain(), interaction(), minBound(), and setMinBound().

6.5.4.3 REAL Domain::xmax[DIM] [private]

domain bounds

Definition at line 53 of file domain.h.

Referenced by boundary(), Domain(), interaction(), maxBound(), and setMaxBound().

6.5.4.4 char Domain::boundaryName[maxBoundaryTypes][WORDLENGTH] [private]

to be depricated

Definition at line 54 of file domain.h.

Referenced by Domain().

6.5.4.5 Boundary Domain::boundaries[DIM][2] [private]

six boundaries of a hexahedral domain

Definition at line 55 of file domain.h.

Referenced by boundary(), BoundaryType(), Domain(), and injection().

6.5.4.6 Collection<Molecule> Domain::molecules [private]

Collection (p. 55) of molecules.

Definition at line 56 of file domain.h.

Referenced by boundary(), computeBounds(), Domain(), injection(), Molecules(), run(), and save().

6.5.4.7 Pool<Molecule>* Domain::pool [private]

Molecule (p. 87) pointer pool.

Definition at line 57 of file domain.h.

Referenced by Domain().

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

- **domain.h**
- **domain.cc**

6.6 Gui::ElementDisp Struct Reference

`#include <gui.h>`

Public Attributes

- int **lighting**
- REAL **rgbcolor** [3]

6.6.1 Detailed Description

Definition at line 97 of file gui.h.

6.6.2 Member Data Documentation

6.6.2.1 int Gui::ElementDisp::lighting

Definition at line 99 of file gui.h.

6.6.2.2 REAL Gui::ElementDisp::rgbcolor[3]

Definition at line 101 of file gui.h.

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

- **gui.h**

6.7 Species::Gas Struct Reference

Gas (p. 77) is a specie with some additional parameters.

`#include <species.h>`

Public Member Functions

- **Gas** (**Specie** *s)

Public Attributes

- REAL **density**
  density in kg/m^3
- **Specie** * **specie**

6.7.1 Detailed Description

Gas (p. 77) is a specie with some additional parameters.

Definition at line 29 of file species.h.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 Species::Gas::Gas (Specie * *s*) [inline]

Definition at line 32 of file species.h.

References specie.

6.7.3 Member Data Documentation

6.7.3.1 REAL Species::Gas::density

density in kg/m^3

Definition at line 30 of file species.h.

Referenced by Domain::Domain(), and Domain::injection().

6.7.3.2 Specie* Species::Gas::specie

Definition at line 31 of file species.h.

Referenced by Gas(), and Domain::injection().

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

- **species.h**

6.8 Item< Element > Struct Template Reference

`#include <collection.h>`

Public Attributes

- Element **element**
- **Item** * **next**
- **Item** * **prev**

6.8.1 Detailed Description

template<class Element> struct Item< Element >

Definition at line 19 of file collection.h.

6.8.2 Member Data Documentation

6.8.2.1 template<class Element> Element Item< Element >::element

Definition at line 20 of file collection.h.

Referenced by Collection< Element >::insert().

6.8.2.2 template<class Element> Item* Item< Element >::next

Definition at line 21 of file collection.h.

Referenced by Collection< Element >::append(), Collection< Element >::insert(), and Collection< Element >::remove().

6.8.2.3 template<class Element> Item * Item< Element >::prev

Definition at line 21 of file collection.h.

Referenced by Collection< Element >::append(), and Collection< Element >::insert().

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

- **collection.h**

6.9 List< Element > Class Template Reference

`#include <list.h>`

Public Member Functions

- **List** ()
- **List** (int **n**)
- **~List** ()
- int **number** ()
- void **setFirst** ()
- void **setFirstLast** ()
- void **goFirst** ()
- void **goFirstNext** ()
- void **FirstNext** ()
- void **FirstPrev** ()
- bool **isFirstLast** ()
- void **goLastNext** ()
- void **goLast** ()
- void **goNext** ()
- void **goPrev** ()
- bool **isFirst** ()
- bool **isLast** ()
- void **LastNext** ()
- void **LastPrev** ()
- void **go** (**Pointer**< Element > *p)
- Element * **First** ()
- Element * **Last** ()
- Element * **Next** ()
- Element * **getNext** ()
- Element * **Prev** ()
- Element * **getPrev** ()
- Element * **Current** ()
- **Pointer**< Element > * **getPointer** ()
- **Pointer**< Element > * **getFirstPointer** ()
- **Pointer**< Element > * **getLastPointer** ()
- int **init** (int **n**)
- int **insert** (Element *element)
- int **insert** ()
- int **append** (Element *element)
- int **append** ()
- int **link** (**Pointer**< Element > *p)
- void **unlink** (**Pointer**< Element > *p)
- int **prepend** (Element *element)
- bool **locate** (Element *element)
- bool **locate** (**Pointer**< Element > *pointer)
- void **moveAfterFirst** ()
- void **swapAfterFirst** ()
- void **moveBehindFirst** ()
- int **erase** ()

- bool **erase** (Element *element)
- void **eraseall** ()
- bool **remove** ()
- void **cleanup** ()
- void **remove** (**Pointer**< Element > *pointer)

Private Attributes

- int **nelements**
- **Pointer**< Element > * **first**
- **Pointer**< Element > * **last**
- **Pointer**< Element > * **current**

6.9.1 Detailed Description

template<class Element> class List< Element >

Definition at line 24 of file list.h.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 template<class Element> List< Element >::List () [inline]

Definition at line 80 of file list.h.

References List< Element >::current, List< Element >::first, List< Element >::last, and List< Element >::nelements.

6.9.2.2 template<class Element> List< Element >::List (int *n*)

6.9.2.3 template<class Element> List< Element >::~List () [inline]

Definition at line 90 of file list.h.

References List< Element >::cleanup().

6.9.3 Member Function Documentation

6.9.3.1 template<class Element> int List< Element >::number () [inline]

Definition at line 31 of file list.h.

6.9.3.2 template<class Element> void List< Element >::setFirst () [inline]

Definition at line 32 of file list.h.

6.9.3.3 template<class Element> void List< Element >::setFirstLast () [inline]

Definition at line 33 of file list.h.

6.9.3.4 template<class Element> void List< Element >::goFirst () [inline]

Definition at line 34 of file list.h.

6.9.3.5 template<class Element> void List< Element >::goFirstNext () [inline]

Definition at line 35 of file list.h.

6.9.3.6 template<class Element> void List< Element >::FirstNext () [inline]

Definition at line 36 of file list.h.

6.9.3.7 template<class Element> void List< Element >::FirstPrev () [inline]

Definition at line 37 of file list.h.

6.9.3.8 template<class Element> bool List< Element >::isFirstLast () [inline]

Definition at line 38 of file list.h.

6.9.3.9 template<class Element> void List< Element >::goLastNext () [inline]

Definition at line 39 of file list.h.

6.9.3.10 template<class Element> void List< Element >::goLast () [inline]

Definition at line 40 of file list.h.

6.9.3.11 template<class Element> void List< Element >::goNext () [inline]

Definition at line 41 of file list.h.

6.9.3.12 template<class Element> void List< Element >::goPrev () [inline]

Definition at line 42 of file list.h.

6.9.3.13 template<class Element> bool List< Element >::isFirst () [inline]

Definition at line 43 of file list.h.

6.9.3.14 template<class Element> bool List< Element >::isLast () [inline]

Definition at line 44 of file list.h.

6.9.3.15 template<class Element> void List< Element >::LastNext () [inline]

Definition at line 45 of file list.h.

6.9.3.16 template<class Element> void List< Element >::LastPrev () [inline]

Definition at line 46 of file list.h.

6.9.3.17 template<class Element> void List< Element >::go (Pointer< Element > * p) [inline]

Definition at line 47 of file list.h.

6.9.3.18 template<class Element> Element* List< Element >::First () [inline]

Definition at line 48 of file list.h.

6.9.3.19 template<class Element> Element* List< Element >::Last () [inline]

Definition at line 49 of file list.h.

6.9.3.20 template<class Element> Element* List< Element >::Next () [inline]

Definition at line 50 of file list.h.

6.9.3.21 template<class Element> Element* List< Element >::getNext () [inline]

Definition at line 51 of file list.h.

6.9.3.22 template<class Element> Element* List< Element >::Prev () [inline]

Definition at line 52 of file list.h.

6.9.3.23 template<class Element> Element* List< Element >::getPrev () [inline]

Definition at line 53 of file list.h.

6.9.3.24 template<class Element> Element* List< Element >::Current () [inline]

Definition at line 54 of file list.h.

6.9.3.25 template<class Element> Pointer<Element>* List< Element >::getPointer () [inline]

Definition at line 55 of file list.h.

6.9.3.26 template<class Element> Pointer<Element>* List< Element >::getFirstPointer () [inline]

Definition at line 56 of file list.h.

6.9.3.27 template<class Element> Pointer<Element>* List< Element >::getLastPointer () [inline]

Definition at line 57 of file list.h.

6.9.3.28 template<class Element> int List< Element >::init (int *n*) [inline]

Definition at line 106 of file list.h.

References List< Element >::append(), i, and List< Element >::nelements.

6.9.3.29 template<class Element> int List< Element >::insert (Element * *element*) [inline]

Definition at line 186 of file list.h.

References List< Element >::current, List< Element >::first, List< Element >::last, List< Element >::nelements, and Pointer< Element >::prev.

6.9.3.30 template<class Element> int List< Element >::insert () [inline]

Definition at line 209 of file list.h.

References Pointer< Element >::element.

6.9.3.31 template<class Element> int List< Element >::append (Element * *element*) [inline]

Definition at line 136 of file list.h.

References List< Element >::current, List< Element >::first, List< Element >::last, List< Element >::nelements, and Pointer< Element >::prev.

6.9.3.32 template<class Element> int List< Element >::append () [inline]

Definition at line 159 of file list.h.

References Pointer< Element >::element.

Referenced by List< Element >::init().

6.9.3.33 template<class Element> int List< Element >::link (Pointer< Element > * *p*) [inline]

Definition at line 165 of file list.h.

References List< Element >::current, List< Element >::first, List< Element >::last, List< Element >::nelements, and Pointer< Element >::prev.

6.9.3.34 template<class Element> void List< Element >::unlink (Pointer< Element > * *p*) [inline]

Definition at line 257 of file list.h.

References List< Element >::current, List< Element >::first, List< Element >::last, List< Element >::nelements, Pointer< Element >::next, and Pointer< Element >::prev.

6.9.3.35 template<class Element> int List< Element >::prepend (Element * *element*) [inline]

Definition at line 113 of file list.h.

References List< Element >::current, List< Element >::first, List< Element >::last, List< Element >::nelements, and Pointer< Element >::next.

6.9.3.36 template<class Element> bool List< Element >::locate (Element * *element*) [inline]

Definition at line 293 of file list.h.

References List< Element >::current, and List< Element >::first.

Referenced by List< Element >::erase(), and List< Element >::remove().

6.9.3.37 template<class Element> bool List< Element >::locate (Pointer< Element > * *pointer*) [inline]

Definition at line 303 of file list.h.

References List< Element >::current, and List< Element >::first.

6.9.3.38 template<class Element> void List< Element >::moveAfterFirst () [inline]

Definition at line 320 of file list.h.

References List< Element >::current, List< Element >::first, and Pointer< Element >::prev.

6.9.3.39 template<class Element> void List< Element >::swapAfterFirst () [inline]

Definition at line 313 of file list.h.

References List< Element >::current, and List< Element >::first.

6.9.3.40 template<class Element> void List< Element >::moveBehindFirst () [inline]

Definition at line 331 of file list.h.

References List< Element >::current, List< Element >::first, and Pointer< Element >::next.

6.9.3.41 template<class Element> int List< Element >::erase () [inline]

Definition at line 235 of file list.h.

References List< Element >::current, Pointer< Element >::element, List< Element >::first, List< Element >::last, and List< Element >::nelements.

Referenced by List< Element >::erase(), and List< Element >::eraseall().

6.9.3.42 template<class Element> bool List< Element >::erase (Element * *element*) [inline]

Definition at line 287 of file list.h.

References List< Element >::erase(), and List< Element >::locate().

6.9.3.43 template<class Element> void List< Element >::eraseall () [inline]

Definition at line 94 of file list.h.

References List< Element >::current, List< Element >::erase(), List< Element >::first, List< Element >::last, and List< Element >::nelements.

6.9.3.44 template<class Element> bool List< Element >::remove () [inline]

Definition at line 215 of file list.h.

References List< Element >::current, List< Element >::first, List< Element >::last, and List< Element >::nelements.

6.9.3.45 template<class Element> void List< Element >::cleanup () [inline]

Definition at line 100 of file list.h.

References List< Element >::current, List< Element >::first, List< Element >::last, and List< Element >::nelements.

Referenced by List< Element >::~List().

6.9.3.46 template<class Element> void List< Element >::remove (Pointer< Element > * *pointer*) [inline]

DDD: slow but safe

DDD

Definition at line 272 of file list.h.

References List< Element >::current, and List< Element >::locate().

6.9.4 Member Data Documentation

6.9.4.1 template<class Element> int List< Element >::nelements [private]

Definition at line 25 of file list.h.

Referenced by List< Element >::append(), List< Element >::cleanup(), List< Element >::erase(), List< Element >::eraseall(), List< Element >::init(), List< Element >::insert(), List< Element

>::link(), List< Element >::List(), List< Gas >::number(), List< Element >::prepend(), List< Element >::remove(), and List< Element >::unlink().

6.9.4.2 template<class Element> struct Pointer< Element >* List< Element >::first [read, private]

Definition at line 26 of file list.h.

Referenced by List< Element >::append(), List< Element >::cleanup(), List< Element >::erase(), List< Element >::eraseall(), List< Gas >::First(), List< Gas >::FirstNext(), List< Gas >::FirstPrev(), List< Gas >::getFirstPointer(), List< Gas >::goFirst(), List< Gas >::goFirstNext(), List< Element >::insert(), List< Gas >::isFirst(), List< Gas >::isFirstLast(), List< Element >::link(), List< Element >::List(), List< Element >::locate(), List< Element >::moveAfterFirst(), List< Element >::moveBehindFirst(), List< Element >::prepend(), List< Element >::remove(), List< Gas >::setFirst(), List< Gas >::setFirstLast(), List< Element >::swapAfterFirst(), and List< Element >::unlink().

6.9.4.3 template<class Element> struct Pointer< Element > * List< Element >::last [read, private]

Definition at line 26 of file list.h.

Referenced by List< Element >::append(), List< Element >::cleanup(), List< Element >::erase(), List< Element >::eraseall(), List< Gas >::getLastPointer(), List< Gas >::goLast(), List< Gas >::goLastNext(), List< Element >::insert(), List< Gas >::isFirstLast(), List< Gas >::isLast(), List< Gas >::Last(), List< Gas >::LastNext(), List< Gas >::LastPrev(), List< Element >::link(), List< Element >::List(), List< Element >::prepend(), List< Element >::remove(), List< Gas >::setFirst(), List< Gas >::setFirstLast(), and List< Element >::unlink().

6.9.4.4 template<class Element> struct Pointer< Element > * List< Element >::current [read, private]

Definition at line 26 of file list.h.

Referenced by List< Element >::append(), List< Element >::cleanup(), List< Gas >::Current(), List< Element >::erase(), List< Element >::eraseall(), List< Gas >::getNext(), List< Gas >::getPointer(), List< Gas >::getPrev(), List< Gas >::go(), List< Gas >::goFirst(), List< Gas >::goFirstNext(), List< Gas >::goLast(), List< Gas >::goLastNext(), List< Gas >::goNext(), List< Gas >::goPrev(), List< Element >::insert(), List< Gas >::isFirst(), List< Gas >::isLast(), List< Element >::link(), List< Element >::List(), List< Element >::locate(), List< Element >::moveAfterFirst(), List< Element >::moveBehindFirst(), List< Gas >::Next(), List< Element >::prepend(), List< Gas >::Prev(), List< Element >::remove(), List< Element >::swapAfterFirst(), and List< Element >::unlink().

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

- **list.h**

6.10 Molecule Class Reference

`#include <model.h>`

Public Member Functions

- **Molecule** ()
- void **Type** (int t)
- int **Type** ()
- void **InternalEnergy** (REAL e)
- REAL **InternalEnergy** ()
- REAL **Coordinate** (int **i**)
- void **Coordinate** (int **i**, REAL **y**)
- REAL * **Coordinates** ()
- void **Coordinates** (REAL **y**[])
- void **setCoordinates** (REAL **y**[])
- REAL **Velocity** (int **i**)
- void **Velocity** (int **i**, REAL u)
- REAL * **Velocity** ()
- void **Velocity** (REAL u[])
- void **setVelocity** (REAL u[])
- REAL **KineticEnergy** ()
- void **Move** ()

  Moving a molecule by one time step.

- void **copy** (**Molecule** *molecule)

Private Attributes

- int **type**

  points to the specie[type];

- REAL **energy**

  internal energy

- REAL **x** [DIM]

  position of center of mass

- REAL **v** [DIM]

  velocity of center of mass

6.10.1 Detailed Description

Definition at line 50 of file model.h.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 Molecule::Molecule ()

Definition at line 19 of file model.cc.

References energy, and type.

6.10.3 Member Function Documentation

6.10.3.1 void Molecule::Type (int *t*) [inline]

Definition at line 67 of file model.h.

References type.

Referenced by Domain::boundary(), copy(), Gui::display(), Boundary::Inject(), Domain::interact(), Domain::interaction(), GridContainer::put(), Domain::run(), and Domain::save().

6.10.3.2 int Molecule::Type () [inline]

Definition at line 68 of file model.h.

References type.

6.10.3.3 void Molecule::InternalEnergy (REAL *e*) [inline]

Definition at line 78 of file model.h.

References energy.

Referenced by Domain::boundary(), Domain::interact(), and Domain::run().

6.10.3.4 REAL Molecule::InternalEnergy () [inline]

Definition at line 79 of file model.h.

References energy.

6.10.3.5 REAL Molecule::Coordinate (int *i*) [inline]

Definition at line 88 of file model.h.

References DIM, and x.

Referenced by copy(), Boundary::Inject(), and Domain::save().

6.10.3.6 void Molecule::Coordinate (int *i*, REAL *y*) [inline]

Definition at line 96 of file model.h.

References DIM, and x.

6.10.3.7 REAL* Molecule::Coordinates () [inline]

Definition at line 103 of file model.h.

References x.

Referenced by Domain::boundary(), Domain::computeBounds(), Gui::display(), Domain::interact(), and GridContainer::put().

6.10.3.8 void Molecule::Coordinates (REAL *y*[ ]) [inline]

Definition at line 104 of file model.h.

References DIM, i, and x.

6.10.3.9 void Molecule::setCoordinates (REAL *y*[ ]) [inline]

Definition at line 105 of file model.h.

References DIM, i, and x.

6.10.3.10 REAL Molecule::Velocity (int *i*) [inline]

Definition at line 106 of file model.h.

References DIM, and v.

Referenced by Domain::boundary(), copy(), Boundary::Inject(), Domain::interact(), and Domain::save().

6.10.3.11 void Molecule::Velocity (int *i*, REAL *u*) [inline]

Definition at line 114 of file model.h.

References DIM, and v.

6.10.3.12 REAL* Molecule::Velocity () [inline]

Definition at line 121 of file model.h.

References v.

6.10.3.13 void Molecule::Velocity (REAL *u*[ ]) [inline]

Definition at line 122 of file model.h.

References DIM, i, and v.

6.10.3.14 void Molecule::setVelocity (REAL *u*[ ]) [inline]

Definition at line 123 of file model.h.

References DIM, i, and v.

6.10.3.15 REAL Molecule::KineticEnergy ()

Definition at line 37 of file model.cc.

References DIM, i, Species::Specie::Mass(), REAL, Species::species, type, and v.

Referenced by Domain::boundary(), and Domain::run().

6.10.3.16 void Molecule::Move ()

Moving a molecule by one time step.

Definition at line 48 of file model.cc.

References DIM, i, Species::Specie::Mass(), REAL, SMALL, Species::species, Run::time, type, v, and x.

Referenced by Domain::run().

6.10.3.17 void Molecule::copy (Molecule * *molecule*)

Definition at line 27 of file model.cc.

References Coordinate(), DIM, i, Type(), type, v, Velocity(), and x.

6.10.4 Member Data Documentation

6.10.4.1 int Molecule::type [private]

points to the specie[type];

Definition at line 51 of file model.h.

Referenced by copy(), KineticEnergy(), Molecule(), Move(), and Type().

6.10.4.2 REAL Molecule::energy [private]

internal energy

Definition at line 58 of file model.h.

Referenced by InternalEnergy(), and Molecule().

6.10.4.3 REAL Molecule::x[DIM] [private]

position of center of mass

Definition at line 58 of file model.h.

Referenced by Coordinate(), Coordinates(), copy(), Move(), and setCoordinates().

6.10.4.4 REAL Molecule::v[DIM] [private]

velocity of center of mass

Definition at line 58 of file model.h.

Referenced by copy(), KineticEnergy(), Move(), setVelocity(), and Velocity().

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

- **model.h**
- **model.cc**

6.11 Option Struct Reference

`#include <run.h>`

Public Attributes

- unsigned int **verbose**:1
- unsigned int **debug**:1
- unsigned int **mesh**:1
- unsigned int **tags**:1
- unsigned int **xterm**:1

6.11.1 Detailed Description

Definition at line 1 of file run.h.

6.11.2 Member Data Documentation

6.11.2.1 unsigned int Option::verbose

Definition at line 2 of file run.h.

Referenced by Run::init(), and Run::readcmdline().

6.11.2.2 unsigned int Option::debug

Definition at line 3 of file run.h.

Referenced by Run::init(), and Run::readcmdline().

6.11.2.3 unsigned int Option::mesh

Definition at line 4 of file run.h.

Referenced by Run::init(), and Run::readcmdline().

6.11.2.4 unsigned int Option::tags

Definition at line 5 of file run.h.

6.11.2.5 unsigned int Option::xterm

Definition at line 6 of file run.h.

Referenced by Domain::Domain(), and Run::init().

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

- **run.h**

6.12 Pointer< Element > Struct Template Reference

`#include <list.h>`

Public Attributes

- Element * **element**
- **Pointer** * **next**
- **Pointer** * **prev**

6.12.1 Detailed Description

template<class Element> struct Pointer< Element >

Definition at line 19 of file list.h.

6.12.2 Member Data Documentation

6.12.2.1 template<class Element> Element* Pointer< Element >::element

Definition at line 20 of file list.h.

Referenced by List< Element >::append(), List< Element >::erase(), and List< Element >::insert().

6.12.2.2 template<class Element> Pointer* Pointer< Element >::next

Definition at line 21 of file list.h.

Referenced by List< Element >::moveBehindFirst(), List< Element >::prepend(), and List< Element >::unlink().

6.12.2.3 template<class Element> Pointer * Pointer< Element >::prev

Definition at line 21 of file list.h.

Referenced by List< Element >::append(), List< Element >::insert(), List< Element >::link(), List< Element >::moveAfterFirst(), and List< Element >::unlink().

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

- **list.h**

6.13 Pool< Element > Struct Template Reference

`#include <collection.h>`

Public Member Functions

- **Pool** (int **nptrs**)
- ~**Pool** ()
- int **size** ()
- **Ptr**< Element > * **get** ()
- bool **put** (**Ptr**< Element > *ptr)
- bool **check** (char *msg)

Public Attributes

- int **mptrs**
- int **nptrs**
- **Ptr**< Element > * **hook**

6.13.1 Detailed Description

template<class Element> struct Pool< Element >

Definition at line 213 of file collection.h.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 template<class Element> Pool< Element >::Pool (int *nptrs*) [inline]

Definition at line 224 of file collection.h.

References Ptr< Element >::element, Pool< Element >::hook, i, Pool< Element >::mptrs, Ptr< Element >::next, Pool< Element >::nptrs, Run::option, and Ptr< Element >::prev.

6.13.2.2 template<class Element> Pool< Element >::~Pool () [inline]

Definition at line 250 of file collection.h.

References Pool< Element >::hook, Pool< Element >::mptrs, and Pool< Element >::nptrs.

6.13.3 Member Function Documentation

6.13.3.1 template<class Element> int Pool< Element >::size () [inline]

Definition at line 218 of file collection.h.

6.13.3.2 template<class Element> Ptr< Element > * Pool< Element >::get () [inline]

DDD: probably not needed, but just to be on the safe side

Definition at line 256 of file collection.h.

References Pool< Element >::hook, Ptr< Element >::next, Pool< Element >::nptrs, and Ptr< Element >::prev.

Referenced by Container< Element >::append(), and Container< Element >::insert().

6.13.3.3 template<class Element> bool Pool< Element >::put (Ptr< Element > * *ptr*) [inline]

Definition at line 270 of file collection.h.

References Pool< Element >::hook, Pool< Element >::mptrs, Ptr< Element >::next, Pool< Element >::nptrs, and Ptr< Element >::prev.

Referenced by Container< Element >::remove().

6.13.3.4 template<class Element> bool Pool< Element >::check (char * *msg*) [inline]

DDD

DDD

DDD

DDD

DDD

DDD

DDD

DDD

Definition at line 294 of file collection.h.

References Pool< Element >::hook, Pool< Element >::mptrs, n, Ptr< Element >::next, and Pool< Element >::nptrs.

Referenced by Container< Element >::checkPool().

6.13.4 Member Data Documentation

6.13.4.1 template<class Element> int Pool< Element >::mptrs

Definition at line 214 of file collection.h.

Referenced by Pool< Element >::check(), Pool< Element >::Pool(), Pool< Element >::put(), and Pool< Element >::~Pool().

6.13.4.2 template<class Element> int Pool< Element >::nptrs

Definition at line 214 of file collection.h.

Referenced by Pool< Element >::check(), Pool< Element >::get(), Container< Element >::number(), Pool< Element >::Pool(), Pool< Element >::put(), and Pool< Element >::~Pool().

6.13.4.3 template<class Element> Ptr<Element>* Pool< Element >::hook

Definition at line 215 of file collection.h.

Referenced by Pool< Element >::check(), Container< Element >::checkPool(), Pool< Element >::get(), Pool< Element >::Pool(), Pool< Element >::put(), and Pool< Element >::~Pool().

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

- **collection.h**

6.14 Ptr< Element > Struct Template Reference

`#include <collection.h>`

Public Attributes

- Element * **element**
- **Ptr** * **next**
- **Ptr** * **prev**

6.14.1 Detailed Description

template<class Element> struct Ptr< Element >

Definition at line 208 of file collection.h.

6.14.2 Member Data Documentation

6.14.2.1 template<class Element> Element* Ptr< Element >::element

Definition at line 209 of file collection.h.

Referenced by Container< Element >::First(), and Pool< Element >::Pool().

6.14.2.2 template<class Element> Ptr* Ptr< Element >::next

Definition at line 210 of file collection.h.

Referenced by Pool< Element >::check(), Container< Element >::checkPool(), Pool< Element >::get(), Container< Element >::insert(), Container< Element >::link(), Pool< Element >::Pool(), Pool< Element >::put(), and Container< Element >::unlink().

6.14.2.3 template<class Element> Ptr * Ptr< Element >::prev

Definition at line 210 of file collection.h.

Referenced by Container< Element >::append(), Pool< Element >::get(), Container< Element >::insert(), Container< Element >::link(), Pool< Element >::Pool(), Pool< Element >::put(), and Container< Element >::unlink().

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

- **collection.h**

6.15 Species::Reaction Struct Reference

Reaction (p. 98) determines the two products only.

`#include <species.h>`

Public Member Functions

- **Reaction** ()
- ~**Reaction** ()
- void **Erase** (**Outcome** *outcome)
- void **Add** (int ip0, int ip1, REAL a, REAL p, REAL h)
- **Outcome** * **First** ()
- **Outcome** * **Next** ()

Public Attributes

- **Species::Reaction::Outcome** * **outcomes**

  Possible outcomes of a reaction.

- **Species::Reaction::Outcome** * **current**

  Possible outcomes of a reaction all outcomes, and the current outcome.

Classes

- struct **Outcome**

  Possible outcomes of a reaction.

6.15.1 Detailed Description

Reaction (p. 98) determines the two products only.

Definition at line 35 of file species.h.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 Species::Reaction::Reaction () [inline]

Definition at line 126 of file species.h.

References current, and outcomes.

6.15.2.2 Species::Reaction::~Reaction () [inline]

Definition at line 127 of file species.h.

References Erase(), and outcomes.

6.15.3 Member Function Documentation

6.15.3.1 void Species::Reaction::Erase (Outcome * *outcome*) [inline]

Definition at line 131 of file species.h.

References Species::Reaction::Outcome::next.

Referenced by ~Reaction().

6.15.3.2 void Species::Reaction::Add (int *ip0*, int *ip1*, REAL *a*, REAL *p*, REAL *h*) [inline]

Add reaction products, probability, and enthalpy:

Definition at line 135 of file species.h.

References Species::Reaction::Outcome::next, and outcomes.

Referenced by Domain::Domain().

6.15.3.3 Outcome* Species::Reaction::First () [inline]

Definition at line 157 of file species.h.

References current, and outcomes.

Referenced by Domain::boundary(), Domain::Domain(), and Domain::interact().

6.15.3.4 Outcome* Species::Reaction::Next () [inline]

Definition at line 158 of file species.h.

References current, Species::Reaction::Outcome::next, and outcomes.

Referenced by Domain::boundary(), and Domain::interact().

6.15.4 Member Data Documentation

6.15.4.1 struct Species::Reaction::Outcome * Species::Reaction::outcomes

Possible outcomes of a reaction.

Referenced by Add(), First(), Next(), Reaction(), and ~Reaction().

6.15.4.2 struct Species::Reaction::Outcome* Species::Reaction::current

Possible outcomes of a reaction all outcomes, and the current outcome.

Referenced by First(), Next(), and Reaction().

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

- **species.h**

6.16 Species::Reaction::Outcome Struct Reference

Possible outcomes of a reaction.

`#include <species.h>`

Public Member Functions

- **Outcome** ()
- **Outcome** (int ip0, int ip1, REAL a, REAL p, REAL h)
- void **Products** (int ip0, int ip1)
- int **Product** (int **i**)
- void **Time** (REAL t)
- REAL **Time** ()
- void **ActivationEnergy** (REAL a)
- REAL **ActivationEnergy** ()
- void **Probability** (REAL r)
- REAL **Probability** ()
- void **Enthalpy** (REAL h)
- REAL **Enthalpy** ()

Public Attributes

- int **product** [2]

  two indexes of the products of a reaction

- REAL **activationEnergy**
- REAL **probability**
- REAL **enthalpy**
- REAL **time**
- **Outcome** ∗ **next**

  linked list of outcomes

6.16.1 Detailed Description

Possible outcomes of a reaction.

Definition at line 37 of file species.h.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 Species::Reaction::Outcome::Outcome () `[inline]`

Definition at line 44 of file species.h.

References enthalpy, next, probability, product, time, and VOIDSPECIE.

6.16.2.2 Species::Reaction::Outcome::Outcome (int *ip0*, int *ip1*, REAL *a*, REAL *p*, REAL *h*) [inline]

Parameters:

ip0 index of the first reaction product

ip1 index of the second reaction product

a activation energy

p outcome probability

h enthalpy of reaction

Definition at line 49 of file species.h.

References activationEnergy, enthalpy, next, Species::nspecies, probability, product, and time.

6.16.3 Member Function Documentation

6.16.3.1 void Species::Reaction::Outcome::Products (int *ip0*, int *ip1*) [inline]

Definition at line 98 of file species.h.

References product.

6.16.3.2 int Species::Reaction::Outcome::Product (int *i*) [inline]

Definition at line 110 of file species.h.

References product.

Referenced by Domain::boundary(), Domain::Domain(), and Domain::interact().

6.16.3.3 void Species::Reaction::Outcome::Time (REAL *t*) [inline]

Definition at line 117 of file species.h.

References time.

Referenced by Domain::boundary().

6.16.3.4 REAL Species::Reaction::Outcome::Time () [inline]

Definition at line 118 of file species.h.

References time.

6.16.3.5 void Species::Reaction::Outcome::ActivationEnergy (REAL *a*) [inline]

Definition at line 119 of file species.h.

References activationEnergy.

Referenced by Domain::interact().

6.16.3.6 REAL Species::Reaction::Outcome::ActivationEnergy () [inline]

Definition at line 120 of file species.h.

References activationEnergy.

6.16.3.7 void Species::Reaction::Outcome::Probability (REAL *r*) [inline]

Definition at line 121 of file species.h.

References probability.

Referenced by Domain::boundary(), Domain::Domain(), and Domain::interact().

6.16.3.8 REAL Species::Reaction::Outcome::Probability () [inline]

Definition at line 122 of file species.h.

References probability.

6.16.3.9 void Species::Reaction::Outcome::Enthalpy (REAL *h*) [inline]

Definition at line 123 of file species.h.

References enthalpy.

Referenced by Domain::boundary(), Domain::Domain(), and Domain::interact().

6.16.3.10 REAL Species::Reaction::Outcome::Enthalpy () [inline]

Definition at line 124 of file species.h.

References enthalpy.

6.16.4 Member Data Documentation

6.16.4.1 int Species::Reaction::Outcome::product[2]

two indexes of the products of a reaction

Definition at line 38 of file species.h.

Referenced by Outcome(), Product(), and Products().

6.16.4.2 REAL Species::Reaction::Outcome::activationEnergy

Definition at line 39 of file species.h.

Referenced by ActivationEnergy(), and Outcome().

6.16.4.3 REAL Species::Reaction::Outcome::probability

Definition at line 39 of file species.h.

Referenced by Outcome(), and Probability().

6.16.4.4 REAL Species::Reaction::Outcome::enthalpy

Definition at line 39 of file species.h.

Referenced by Enthalpy(), and Outcome().

6.16.4.5 REAL Species::Reaction::Outcome::time

time of reaction: used primarily for surface reactions, where there is a time delay between species adsorption on the surface and product formation.

Definition at line 39 of file species.h.

Referenced by Outcome(), and Time().

6.16.4.6 Outcome* Species::Reaction::Outcome::next

linked list of outcomes

Definition at line 43 of file species.h.

Referenced by Species::Reaction::Add(), Species::Reaction::Erase(), Species::Reaction::Next(), and Outcome().

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

- **species.h**

6.17 Gui::Scene Struct Reference

`#include <gui.h>`

Public Attributes

- **Gui::Scene::Color color**
- **Gui::Scene::Mesh mesh**
- **Gui::Scene::Frame frame**

Classes

- struct **Color**
- struct **Frame**
- struct **Mesh**

6.17.1 Detailed Description

Definition at line 107 of file gui.h.

6.17.2 Member Data Documentation

6.17.2.1 struct Gui::Scene::Color Gui::Scene::color

Referenced by Gui::display(), Gui::init(), Gui::readconf(), and Gui::switchColorScheme().

6.17.2.2 struct Gui::Scene::Mesh Gui::Scene::mesh

Referenced by Gui::display(), Gui::initdisp(), and Gui::readconf().

6.17.2.3 struct Gui::Scene::Frame Gui::Scene::frame

Referenced by Gui::display(), Gui::initdisp(), and Gui::readconf().

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

- **gui.h**

6.18 Gui::Scene::Color Struct Reference

`#include <gui.h>`

Public Attributes

- **ColorSchemes scheme**
- REAL **minvalue**
- REAL **maxvalue**

6.18.1 Detailed Description

Definition at line 108 of file gui.h.

6.18.2 Member Data Documentation

6.18.2.1 ColorSchemes Gui::Scene::Color::scheme

Definition at line 109 of file gui.h.

Referenced by Gui::display(), Gui::init(), Gui::readconf(), and Gui::switchColorScheme().

6.18.2.2 REAL Gui::Scene::Color::minvalue

Definition at line 110 of file gui.h.

Referenced by Gui::display(), and Gui::readconf().

6.18.2.3 REAL Gui::Scene::Color::maxvalue

Definition at line 110 of file gui.h.

Referenced by Gui::display(), and Gui::readconf().

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

- **gui.h**

6.19 Gui::Scene::Frame Struct Reference

`#include <gui.h>`

Public Attributes

- REAL **line** [maxlineprm]

6.19.1 Detailed Description

Definition at line 117 of file gui.h.

6.19.2 Member Data Documentation

6.19.2.1 REAL Gui::Scene::Frame::line[maxlineprm]

Definition at line 119 of file gui.h.

Referenced by Gui::display(), Gui::initdisp(), and Gui::readconf().

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

- **gui.h**

6.20 Gui::Scene::Mesh Struct Reference

`#include <gui.h>`

Public Attributes

- REAL **node** [maxpointprm]
- REAL **line** [maxlineprm]

6.20.1 Detailed Description

Definition at line 112 of file gui.h.

6.20.2 Member Data Documentation

6.20.2.1 REAL Gui::Scene::Mesh::node[maxpointprm]

Definition at line 114 of file gui.h.

Referenced by Gui::display(), Gui::initdisp(), and Gui::readconf().

6.20.2.2 REAL Gui::Scene::Mesh::line[maxlineprm]

Definition at line 114 of file gui.h.

Referenced by Gui::initdisp(), and Gui::readconf().

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

- **gui.h**

6.21 Species::Specie Class Reference

`#include <species.h>`

Public Member Functions

- **Specie** ()
- void **Cp** (REAL c)
- REAL **Cp** ()
- char * **Id** ()
- void **Mass** (REAL m)
- REAL **Mass** ()
- REAL **Size** ()
- void **Size** (REAL s)

Private Attributes

- char **id** [WORDLENGTH]
- REAL **mass**
- REAL **size**
- REAL **cp**
- REAL **inertia** [DIM]

6.21.1 Detailed Description

Definition at line 6 of file species.h.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 Species::Specie::Specie () [inline]

Definition at line 13 of file species.h.

References cp, DIM, i, inertia, mass, and size.

6.21.3 Member Function Documentation

6.21.3.1 void Species::Specie::Cp (REAL *c*) [inline]

Definition at line 20 of file species.h.

References cp.

Referenced by Domain::boundary(), and Domain::interact().

6.21.3.2 REAL Species::Specie::Cp () [inline]

Definition at line 21 of file species.h.

References cp.

6.21.3.3 char* Species::Specie::Id () [inline]

Definition at line 22 of file species.h.

References id.

Referenced by Domain::Domain(), Domain::interact(), and Domain::save().

6.21.3.4 void Species::Specie::Mass (REAL *m*) [inline]

Definition at line 23 of file species.h.

References mass.

Referenced by Domain::boundary(), Gui::display(), Domain::Domain(), Domain::injection(), Domain::interact(), Molecule::KineticEnergy(), and Molecule::Move().

6.21.3.5 REAL Species::Specie::Mass () [inline]

Definition at line 24 of file species.h.

References mass.

6.21.3.6 REAL Species::Specie::Size () [inline]

Definition at line 25 of file species.h.

References size.

Referenced by Domain::boundary(), Gui::display(), Domain::Domain(), Domain::interact(), Domain::run(), and Domain::save().

6.21.3.7 void Species::Specie::Size (REAL *s*) [inline]

Definition at line 26 of file species.h.

References size.

6.21.4 Member Data Documentation

6.21.4.1 char Species::Specie::id[WORDLENGTH] [private]

Definition at line 7 of file species.h.

Referenced by Id().

6.21.4.2 REAL Species::Specie::mass [private]

Definition at line 8 of file species.h.

Referenced by Mass(), and Specie().

6.21.4.3 REAL Species::Specie::size [private]

Definition at line 8 of file species.h.

Referenced by Size(), and Specie().

6.21.4.4 REAL Species::Specie::cp [private]

Definition at line 8 of file species.h.

Referenced by Cp(), and Specie().

6.21.4.5 REAL Species::Specie::inertia[DIM] [private]

Definition at line 8 of file species.h.

Referenced by Specie().

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

- **species.h**

6.22 Run::Time Struct Reference

`#include <run.h>`

Public Attributes

- double **start**
- double **end**
- double **prev**
- double **current**
- double **step**
- double **step0**
- double **output**

6.22.1 Detailed Description

Definition at line 18 of file run.h.

6.22.2 Member Data Documentation

6.22.2.1 double Run::Time::start

Definition at line 19 of file run.h.

6.22.2.2 double Run::Time::end

Definition at line 19 of file run.h.

6.22.2.3 double Run::Time::prev

Definition at line 19 of file run.h.

6.22.2.4 double Run::Time::current

Definition at line 19 of file run.h.

6.22.2.5 double Run::Time::step

Definition at line 19 of file run.h.

6.22.2.6 double Run::Time::step0

Definition at line 19 of file run.h.

6.22.2.7 double Run::Time::output

Definition at line 19 of file run.h.

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

- **run.h**

6.23 Gui::WindowGeom Struct Reference

`#include <gui.h>`

Public Attributes

- int **width**
- int **height**

6.23.1 Detailed Description

Definition at line 169 of file gui.h.

6.23.2 Member Data Documentation

6.23.2.1 int Gui::WindowGeom::width

Definition at line 172 of file gui.h.

Referenced by Gui::init(), and Gui::initdisp().

6.23.2.2 int Gui::WindowGeom::height

Definition at line 172 of file gui.h.

Referenced by Gui::init(), and Gui::initdisp().

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

- **gui.h**

Chapter 7

ReMoDy File Documentation

7.1 collection.h File Reference

Classes

- struct **Item< Element >**
- class **Collection< Element >**
- struct **Ptr< Element >**
- struct **Pool< Element >**
- class **Container< Element >**

7.2 container.h File Reference

7.3 def.h File Reference

Defines

- #define **ABOUT** "MOLECULAR DYNAMICS SIMULATOR BY A.V.SMIRNOV\nandrei.v.smirnov@gmail.com**n**"
- #define **DEBUG**
- #define **DOCTYPE** "remody"
- #define **REAL** double
- #define **TINY** 1e-10
- #define **SMALL** 1e-30
- #define **LARGE** 1e30
- #define **MAXLINLEN** 510
- #define **WORDLENGTH** 64
- #define **ESC** 0x1B
- #define **PI** 3.14159265358979323846
- #define **SQRT2** 1.4142
- #define **SQRTPIo8** 0.62666
- #define **ERROR**(message) { fprintf(stderr,"\nERROR: %s**n**",message);exit(1);}
- #define **TRUE** 1
- #define **FALSE** 0
- #define **LOCAL**
- #define **DIM** 3
- #define **BoltzmannConstant** 1.38066e-23
- #define **AtomicMassUnit** 1.66053886e-27
- #define **AvogadroNumber** 6.02214179e23
- #define **RND** (REAL)((double)rand()/RAND_MAX)
- #define **ZERO**(**n**, x) for (int **i**=0; **i**<(**n**); **i**++)x[**i**]=0.0
- #define **ADD**(**n**, x, **y**) for (int **i**=0; **i**<(**n**); **i**++) (x)[**i**] += (**y**)[**i**]
- #define **SUB**(**n**, x, **y**) for (int **i**=0; **i**<(**n**); **i**++) (x)[**i**] -= (**y**)[**i**]
- #define **ADDC**(**n**, x, **y**, d) for (int **i**=0; **i**<(**n**); **i**++) (x)[**i**] += (**y**)[**i**]∗(d)
- #define **SUBC**(**n**, x, **y**, d) for (int **i**=0; **i**<(**n**); **i**++) (x)[**i**] -= (**y**)[**i**]∗(d)
- #define **MUL**(**n**, x, **y**) for (int **i**=0; **i**<(**n**); **i**++) (x)[**i**] ∗= (**y**)[**i**]
- #define **MULC**(**n**, x, c) for (int **i**=0; **i**<(**n**); **i**++) (x)[**i**] ∗= (c)
- #define **DIV**(**n**, x, **y**) for (int **i**=0; **i**<(**n**); **i**++) (x)[**i**] /= (**y**)[**i**]
- #define **MULVEC**(x, s) for (int **i**=0; **i**<DIM; **i**++) (x)[**i**] ∗= (s)
- #define **ADDVEC**(x, **y**) for (int **i**=0; **i**<DIM; **i**++) (x)[**i**] += (**y**)[**i**]
- #define **COPYVEC**(x, **y**) for (int **i**=0; **i**<DIM; **i**++) (x)[**i**] = (**y**)[**i**]
- #define **ZEROVEC**(x) for (int **i**=0; **i**<DIM; **i**++) (x)[**i**] = 0.
- #define **SCLP**(A, B) (A[0]∗B[0]+A[1]∗B[1]+A[2]∗B[2])
- #define **VECP**(A, B, C)
- #define **LENGTH**(A) sqrt(SCLP(A,A))
- #define **VOIDSPECIE** nspecies

7.3.1 Define Documentation

7.3.1.1 #define ABOUT "MOLECULAR DYNAMICS SIMULATOR BY A.V.SMIRNOV\nandrei.v.smirnov@gmail.com\n"

Definition at line 1 of file def.h.

Referenced by Gui::menu().

7.3.1.2 #define ADD(n, x, y) for (int i=0; i<(n); i++) (x)[i] += (y)[i]

Definition at line 29 of file def.h.

7.3.1.3 #define ADDC(n, x, y, d) for (int i=0; i<(n); i++) (x)[i] += (y)[i]*(d)

Definition at line 31 of file def.h.

7.3.1.4 #define ADDVEC(x, y) for (int i=0; i<DIM; i++) (x)[i] += (y)[i]

Definition at line 37 of file def.h.

7.3.1.5 #define AtomicMassUnit 1.66053886e-27

Definition at line 21 of file def.h.

Referenced by Domain::Domain(), Domain::injection(), and Domain::run().

7.3.1.6 #define AvogadroNumber 6.02214179e23

Definition at line 22 of file def.h.

Referenced by Domain::Domain().

7.3.1.7 #define BoltzmannConstant 1.38066e-23

Definition at line 20 of file def.h.

Referenced by Domain::Domain(), Domain::injection(), and Domain::run().

7.3.1.8 #define COPYVEC(x, y) for (int i=0; i<DIM; i++) (x)[i] = (y)[i]

Definition at line 38 of file def.h.

7.3.1.9 #define DEBUG

Definition at line 2 of file def.h.

7.3.1.10 #define DIM 3

Definition at line 19 of file def.h.

Referenced by Geom::area(), Domain::BoundaryType(), Molecule::Coordinate(), Molecule::Coordinates(), Molecule::copy(), Geom::distance(), Geom::distvec(), Domain::Domain(), GridContainer::index(), GridContainer::init(), Gui::initdisp(), Boundary::Inject(), Molecule::KineticEnergy(), Geom::length(), Molecule::Move(), Geom::normalize(), GridContainer::put(), Gui::readconf(), Geom::sclp(), Molecule::setCoordinates(), Molecule::setVelocity(), Species::Specie::Specie(), and Molecule::Velocity().

7.3.1.11 #define DIV(n, x, y) for (int i=0; i<(n); i++) (x)[i] /= (y)[i]

Definition at line 35 of file def.h.

7.3.1.12 #define DOCTYPE "remody"

Definition at line 3 of file def.h.

Referenced by Domain::Domain(), IO::getIter(), IO::getTimeXML(), and Gui::readconf().

7.3.1.13 #define ERROR(message) { fprintf(stderr,"\nERROR: %s\n",message);exit(1);}

Definition at line 14 of file def.h.

Referenced by Gui::init().

7.3.1.14 #define ESC 0x1B

Definition at line 10 of file def.h.

Referenced by Gui::animate(), Gui::keyboard(), main(), and Domain::run().

7.3.1.15 #define FALSE 0

Definition at line 17 of file def.h.

7.3.1.16 #define LARGE 1e30

Definition at line 7 of file def.h.

Referenced by Domain::computeBounds(), and Gui::initdisp().

7.3.1.17 #define LENGTH(A) sqrt(SCLP(A,A))

Definition at line 44 of file def.h.

Referenced by Geom::area(), Domain::boundary(), and Domain::run().

7.3.1.18 #define LOCAL

Definition at line 18 of file def.h.

7.3.1.19 #define MAXLINLEN 510

Definition at line 8 of file def.h.

Referenced by Gui::commandMode(), Gui::consoleMenu(), Domain::Domain(), Gui::Exit(), IO::getCharAttr(), IO::getIntAttr(), IO::getIter(), IO::getTimeXML(), Gui::keyboard(), IO::parseWord(), Gui::Quit(), Gui::readconf(), Domain::save(), and IO::xwd().

7.3.1.20 #define MUL(n, x, y) for (int i=0; i<(n); i++) (x)[i] *= (y)[i]

Definition at line 33 of file def.h.

7.3.1.21 #define MULC(n, x, c) for (int i=0; i<(n); i++) (x)[i] *= (c)

Definition at line 34 of file def.h.

Referenced by Domain::boundary().

7.3.1.22 #define MULVEC(x, s) for (int i=0; i<DIM; i++) (x)[i] *= (s)

Definition at line 36 of file def.h.

7.3.1.23 #define PI 3.14159265358979323846

Definition at line 11 of file def.h.

Referenced by Domain::injection(), Gui::showParticle(), and Gui::showSphere().

7.3.1.24 #define REAL double

Definition at line 4 of file def.h.

Referenced by Geom::area(), Domain::boundary(), Domain::computeBounds(), Gui::display(), Geom::distance(), Domain::Domain(), GridContainer::init(), Gui::initdisp(), Boundary::Inject(), Domain::injection(), Domain::interact(), Domain::interaction(), Molecule::KineticEnergy(), Geom::length(), Gui::motion(), Molecule::Move(), Geom::normalize(), IO::parseFloat(), Palette::pickcolor(), GridContainer::put(), Gui::readconf(), Run::rnd(), Domain::run(), Geom::sclp(), Gui::showParticle(), Gui::showSphere(), Run::testrnd(), and Potential::value().

7.3.1.25 #define RND (REAL)((double)rand()/RAND_MAX)

Definition at line 24 of file def.h.

Referenced by Domain::boundary(), Boundary::Inject(), Domain::injection(), and Domain::interact().

7.3.1.26 #define SCLP(A, B) (A[0]*B[0]+A[1]*B[1]+A[2]*B[2])

Definition at line 40 of file def.h.

Referenced by Domain::interact().

7.3.1.27 #define SMALL 1e-30

Definition at line 6 of file def.h.

Referenced by Domain::boundary(), Domain::interact(), Molecule::Move(), and Domain::run().

7.3.1.28 #define SQRT2 1.4142

Definition at line 12 of file def.h.

7.3.1.29 #define SQRTPIo8 0.62666

Definition at line 13 of file def.h.

7.3.1.30 #define SUB(n, x, y) for (int i=0; i<(n); i++) (x)[i] -= (y)[i]

Definition at line 30 of file def.h.

7.3.1.31 #define SUBC(n, x, y, d) for (int i=0; i<(n); i++) (x)[i] -= (y)[i]*(d)

Definition at line 32 of file def.h.

7.3.1.32 #define TINY 1e-10

Definition at line 5 of file def.h.

Referenced by Domain::Domain().

7.3.1.33 #define TRUE 1

Definition at line 16 of file def.h.

7.3.1.34 #define VECP(A, B, C)

Value:

```
A[0]=B[1]*C[2]-B[2]*C[1];\
                                                A[1]=B[2]*C[0]-B[0]*C[2];\
                                                A[2]=B[0]*C[1]-B[1]*C[0];
```

Definition at line 41 of file def.h.

Referenced by Geom::area().

7.3.1.35 #define VOIDSPECIE nspecies

Definition at line 45 of file def.h.

Referenced by Domain::boundary(), Domain::Domain(), Domain::interact(), Domain::interaction(), Species::Reaction::Outcome::Outcome(), GridContainer::put(), Domain::run(), and Domain::save().

7.3.1.36 #define WORDLENGTH 64

Definition at line 9 of file def.h.

Referenced by Domain::Domain(), and main().

7.3.1.37 #define ZERO(n, x) for (int i=0; i<(n); i++)x[i]=0.0

Definition at line 27 of file def.h.

7.3.1.38 #define ZEROVEC(x) for (int i=0; i<DIM; i++) (x)[i] = 0.

Definition at line 39 of file def.h.

7.4 doc.h File Reference

7.5 domain.cc File Reference

```
#include <time.h>
#include <iostream>
#include <libxml/xmlmemory.h>
#include <libxml/parser.h>
#include <zlib.h>
#include <math.h>
#include "def.h"
#include "io.h"
#include "run.h"
#include "list.h"
#include "collection.h"
#include "grid.h"
#include "container.h"
#include "model.h"
#include "species.h"
#include "domain.h"
#include "gui.h"
```

Namespaces

- namespace **std**

Variables

- const REAL **GasConstant** = BoltzmannConstant∗AvogadroNumber

7.5.1 Variable Documentation

7.5.1.1 const REAL GasConstant = BoltzmannConstant∗AvogadroNumber

Definition at line 21 of file domain.cc.

Referenced by Domain::boundary(), Domain::Domain(), and Domain::interact().

7.6 domain.h File Reference

Classes

- class **Boundary**
  Domain *(p. 70)* ***Boundary*** *(p. 51) Definition.*
- class **Domain**
  Computational ***Domain*** *(p. 70).*

Enumerations

- enum **BoundaryTypes** {
  insideBoundary = 0, **periodicBoundary**, **elasticBoundary**, **openBoundary**,
  maxBoundaryTypes }

7.6.1 Detailed Description

Main definitions of the **Domain** (p. 70) class and its associate class **Boundary** (p. 51)
Definition in file **domain.h**.

7.6.2 Enumeration Type Documentation

7.6.2.1 enum BoundaryTypes

Enumerator:

insideBoundary
periodicBoundary
elasticBoundary
openBoundary
maxBoundaryTypes

Definition at line 5 of file domain.h.

7.7 grid.cc File Reference

```
#include <math.h>
#include <iostream>
#include "def.h"
#include "run.h"
#include "list.h"
#include "collection.h"
#include "species.h"
#include "grid.h"
#include "container.h"
#include "model.h"
```

Namespaces

- namespace **GridContainer**

Functions

- void **GridContainer::init** (REAL ymin[ ], REAL ymax[ ], REAL radius, **Pool**< **Molecule** > ∗newpool)
- int **GridContainer::index** (int ind[ ])
- void **GridContainer::index** (int **n**, int ind[ ])
- void **GridContainer::put** (**Molecule** ∗node)
- **Container**< **Molecule** > ∗ **GridContainer::get** (int ind[ ])
- int ∗ **GridContainer::dimensions** ()

Variables

- REAL **GridContainer::xmin** [DIM]
- REAL **GridContainer::xmax** [DIM]
- REAL **GridContainer::cellsize**
- int **GridContainer::ncells** [DIM+1]
- int **GridContainer::mcells** = 0
- int **GridContainer::mcells1** = 0
- **Container**< **Molecule** > ∗ **GridContainer::nodes**
- **Pool**< **Molecule** > ∗ **GridContainer::pool**

7.8 grid.h File Reference

Namespaces

- namespace **GridContainer**

Functions

- int * **GridContainer::dimensions** ()
- int **GridContainer::index** (int ind[])
- void **GridContainer::index** (int **n**, int ind[])
- void **GridContainer::init** (REAL ymin[], REAL ymax[], REAL radius, **Pool**< **Molecule** > *newpool)
- void **GridContainer::put** (**Molecule** *node)
- **Container**< **Molecule** > * **GridContainer::get** (int ind[])
- bool **GridContainer::checkPool** (int icell, char *msg)

Variables

- REAL **GridContainer::cellsize**
- int **GridContainer::ncells** [DIM+1]
- **Container**< **Molecule** > * **GridContainer::nodes**
- **Pool**< **Molecule** > * **GridContainer::pool**

7.9 gui.cc File Reference

```
#include <iostream>
#include <fstream>
#include <math.h>
#include <string.h>
#include <libxml/xmlmemory.h>
#include <GL/glut.h>
#include <GL/glx.h>
#include <GL/glu.h>
#include <X11/keysym.h>
#include "def.h"
#include "run.h"
#include "io.h"
#include "list.h"
#include "collection.h"
#include "model.h"
#include "species.h"
#include "domain.h"
#include "gui.h"
```

Namespaces

- namespace **Gui**
- namespace **Palette**

Functions

- int **Gui::query_extension** (char *extName)
- void **Gui::init** (int argc, char *argv[], **Domain** *newdomain)
- void **Gui::helpDisplay** ()
- int **Gui::readparam** (char *s, char *param[], int maxparam, char *filename, int val[])
- int **Gui::readparam** (char *s, char *param[], int maxparam, char *filename, REAL val[])
- void **Gui::readconf** ()
- void **Gui::refresh** ()
- void **Gui::initdisp** ()
- void **Gui::Materials** (int argc, char *argv[])
- void **Gui::showVector** (double *x, double *v)
- void **Gui::showParticle** (double vmn, double vmx, double val, double *x)
- void **Gui::showSphere** (double vmn, double vmx, double val, double rad, double *x)
- void **Gui::displayAxes** ()
- void **Gui::getScaling** (double &l0, double &l1)
- void **Gui::getXLimits** (double *x0, double *x1)

- void **Gui::Exit** ()
- void **Gui::Quit** ()
- void **Gui::menu** (int value)
- void **Gui::displayMenu** ()
- void **Gui::consoleMenu** ()
- void **Gui::setBackgroundRun** ()
- void **Gui::setForegroundRun** ()
- void **Gui::toggleSpheres** ()
- void **Gui::switchColorScheme** ()
- void **Gui::runmany** ()
- void **Gui::toggleWindowDump** ()
- void **Gui::dumpwindow** ()
- void **Gui::commandMode** ()
- void **Gui::display** ()
- void **Gui::animate** ()
- void **Gui::helpCommand** ()
- void **Gui::reshape** (int w, int h)
- void **Gui::mouse** (int button, int state, int x, int **y**)
- void **Gui::motion** (int x, int **y**)
- void **Gui::keyboard** (unsigned int key)
- void **Gui::run** ()
- void **Gui::displaymessage** (char *msg)
- void **Gui::setElementColor** (int element_type, REAL rgbcolor[])
- void **Gui::getElementColor** (int element_type, REAL rgbcolor[])
- void **Palette::init** (REAL vmin, REAL vmax)
- void **Palette::pickcolor** (REAL var, REAL color[])

Variables

- **Domain** * **Gui::domain**
- ElementDisp **Gui::disp** [maxelements]
- char * **Gui::configfile** = "gui.cfg"
- char **Gui::windowname** [MAXLINLEN]
- int **Gui::showpar** [maxshowpars]
- int **Gui::finished**
- int **Gui::animation**
- int **Gui::nwdump**
- int **Gui::iwdump**
- int **Gui::attributeList** [ ]
- REAL **Gui::step**
- REAL **Gui::vecval** [maxlineprm]
- REAL **Gui::axes** [maxaxesprm]
- REAL **Gui::rgbcolor** [maxcolor][3]
- REAL **Gui::wdtime**
- REAL **Gui::xo** [3]
- REAL **Gui::dx**
- REAL **Gui::dy**
- REAL **Gui::dz**
- REAL **Gui::lastx**
- REAL **Gui::lasty**

- REAL **Gui::lastz**
- int **Gui::mouseButtons** [3]
- REAL **Gui::zoom**
- REAL **Gui::rotx**
- REAL **Gui::roty**
- REAL **Gui::tx**
- REAL **Gui::ty**
- REAL **Gui::xmin** [DIM]
- REAL **Gui::xmax** [DIM]
- REAL **Gui::lx**
- REAL **Gui::ly**
- REAL **Gui::lz**
- REAL **Gui::lmin**
- REAL **Gui::lmax**
- enum Movement **Gui::movement**
- ColorScale **Gui::colorscale**
- Scene **Gui::scene**
- WindowGeom **Gui::window**
- BFaceList * **Gui::bface_root**
- Display * **Gui::dpy**
- Window **Gui::win**
- int **Gui::firstR** = 1
- int * **Gui::vars**
- int * **Gui::coms**
- REAL **Palette::vmn**
- REAL **Palette::vmx**
- REAL **Palette::vav**
- REAL **Palette::dvi**

7.10 gui.h File Reference

Namespaces

- namespace **Gui**
- namespace **Palette**

Classes

- struct **Gui::ElementDisp**
- struct **Gui::ColorScale**
- struct **Gui::Scene**
- struct **Gui::Scene::Color**
- struct **Gui::Scene::Mesh**
- struct **Gui::Scene::Frame**
- struct **Gui::WindowGeom**

Defines

- #define **WINDOW_SIZE** 800
- #define **MAX_WINDOW_WIDTH** 800
- #define **MAX_WINDOW_HEIGHT** 600
- #define **LEFT_BUTTON** 1
- #define **MIDDLE_BUTTON** 2
- #define **RIGHT_BUTTON** 3

Enumerations

- enum **Gui::Elements** {
  Gui::points = 0, **Gui::nodes**, **Gui::edges**, **Gui::faces**,
  Gui::cells, **Gui::frame**, **Gui::mesh**, **Gui::boundary_nodes**,
  Gui::boundary_edges, **Gui::boundary_faces**, **Gui::boundary_cells**, **Gui::maxelements** }
- enum **Gui::Color** {
  Gui::red = 0, **Gui::green**, **Gui::blue**, **Gui::skyblue**,
  Gui::brown, **Gui::magenta**, **Gui::yellow**, **Gui::color6**,
  Gui::color7, **Gui::color8**, **Gui::color9**, **Gui::color10**,
  Gui::color11, **Gui::color12**, **Gui::color13**, **Gui::color14**,
  Gui::color15, **Gui::maxcolor** }
- enum **Gui::ColorSchemes** {
  Gui::colorByType = 0, **Gui::colorByBoundary**, **Gui::colorByTime**, **Gui::colorByMass**,
  Gui::maxColorSchemes }
- enum **Gui::PointParameters** {
  Gui::variable = maxcolor, **Gui::vmin**, **Gui::vmax**, **Gui::size**,
  Gui::sizevar, **Gui::massvar**, **Gui::maxpointprm** }

- enum **Gui::LineParameters** { **Gui::thickness** = maxcolor, **Gui::length**, **Gui::maxlineprm** }
- enum **Gui::AxesParameters** {
  Gui::axeswidth = 0, **Gui::xaxislength**, **Gui::yaxislength**, **Gui::zaxislength**,
  Gui::arrowheight, **Gui::arrowidth**, **Gui::maxaxesprm** }
- enum **Gui::SurfDispType** { **Gui::gridlines** = 0, **Gui::solidsurface**, **Gui::maxsurfdisptypes** }
- enum **Gui::Movement** { **Gui::stay** = 0, **Gui::rotate**, **Gui::moveuv**, **Gui::movew** }
- enum **Gui::ShowPars** {
  Gui::showRun = 0, **Gui::showAxes**, **Gui::showSpheres**, **Gui::showBonds**,
  Gui::showGrid, **Gui::showNodes**, **Gui::showVariables**, **Gui::showBoundaryVertexes**,
  Gui::showBoundaryVectors, **Gui::showToolVertexes**, **Gui::showBoundaryFaceCenters**, **Gui::showBoundaryFaces**,
  Gui::showBoundaryGrid, **Gui::showToolGrid**, **Gui::showFrame**, **Gui::showCellCenters**,
  Gui::showFaceCenters, **Gui::showIsoSurfaces**, **Gui::dumpWindow**, **Gui::maxshowpars** }

Functions

- void **Gui::Exit** ()
- void **Gui::Quit** ()
- void **Gui::readconf** ()
- void **Gui::helpDisplay** ()
- int **Gui::readparam** (char *s, char *param[], int maxparam, char *filename, REAL val[])
- void **Gui::initdisp** ()
- void **Gui::Materials** (int argc, char *argv[])
- void **Gui::display** ()
- void **Gui::displayAxes** ()
- void **Gui::writeData** ()
- void **Gui::helpCommand** ()
- void **Gui::printGridXLimits** ()
- void **Gui::printGridVecLimits** ()
- void **Gui::getXLimits** (REAL *xmin, REAL *xmax)
- void **Gui::getScaling** (REAL &lmin, REAL &lmax)
- void **Gui::printPartVelLimits** ()
- void **Gui::printParticleVariables** ()
- void **Gui::showGridElements** (int ne, double *X, REAL color[])
- void **Gui::showVectorComponent** (int ivar, int icomp, double *X, double vmn, double vmx)
- void **Gui::showVector** (int ivar, double *X)
- void **Gui::commandMode** ()
- void **Gui::printVariables** (int **n**)
- void **Gui::init** (int argc, char *argv[], **Domain** *newdomain)
- void **Gui::finish** ()
- void **Gui::InitMaterials** (void)
- void **Gui::getScaling** (double &l0, double &l1)
- void **Gui::getXLimits** (double *x0, double *x1)

- void **Gui::selectVariable** ()
- void **Gui::setBackgroundRun** ()
- void **Gui::setForegroundRun** ()
- void **Gui::animate** ()
- void **Gui::reshape** (int w, int h)
- void **Gui::mouse** (int button, int state, int x, int **y**)
- void **Gui::motion** (int x, int **y**)
- void **Gui::keyboard** (unsigned char key, int x, int **y**)
- void **Gui::menu** (int value)
- void **Gui::run** ()
- void **Gui::drawSegment** (REAL *x, REAL *__y__)
- void **Gui::displaymessage** (char *msg)
- void **Gui::setElementColor** (int element_type, REAL rgbcolor[])
- void **Gui::getElementColor** (int element_type, REAL rgbcolor[])
- void **Palette::init** (REAL vmin, REAL vmax)
- void **Palette::pickcolor** (REAL var, REAL color[])

Variables

- char * **Gui::configfile**
- REAL **Gui::lx**
- REAL **Gui::ly**
- REAL **Gui::lz**
- REAL **Gui::lmin**
- REAL **Gui::lmax**
- REAL **Gui::step**
- REAL **Gui::vecval** [maxlineprm]
- REAL **Gui::axes** [maxaxesprm]
- REAL **Gui::rgbcolor** [maxcolor][3]
- ElementDisp **Gui::disp** [maxelements]
- int **Gui::showpar** [maxshowpars]
- int **Gui::finished**
- int **Gui::animation**
- REAL **Gui::dx**
- REAL **Gui::dy**
- REAL **Gui::dz**
- REAL **Gui::lastx**
- REAL **Gui::lasty**
- REAL **Gui::lastz**
- int **Gui::mouseButtons** [3]
- REAL **Gui::zoom**
- REAL **Gui::rotx**
- REAL **Gui::roty**
- REAL **Gui::tx**
- REAL **Gui::ty**

7.10.1 Define Documentation

7.10.1.1 #define LEFT_BUTTON 1

Definition at line 10 of file gui.h.

Referenced by Gui::mouse().

7.10.1.2 #define MAX_WINDOW_HEIGHT 600

Definition at line 8 of file gui.h.

Referenced by Gui::initdisp().

7.10.1.3 #define MAX_WINDOW_WIDTH 800

Definition at line 7 of file gui.h.

Referenced by Gui::initdisp().

7.10.1.4 #define MIDDLE_BUTTON 2

Definition at line 11 of file gui.h.

Referenced by Gui::mouse().

7.10.1.5 #define RIGHT_BUTTON 3

Definition at line 12 of file gui.h.

Referenced by Gui::mouse().

7.10.1.6 #define WINDOW_SIZE 800

Definition at line 6 of file gui.h.

Referenced by Gui::initdisp().

7.11 io.cc File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <ctype.h>
#include <libxml/xmlmemory.h>
#include <libxml/parser.h>
#include "def.h"
#include "run.h"
#include "io.h"
```

Namespaces

- namespace **IO**

Functions

- void **IO::initxwd** (int **i**)
- void **IO::xwd** (char *windowname)
- int **IO::getCharAttr** (xmlNodePtr cur, char *attr, char *result)
- int **IO::getIntAttr** (xmlNodePtr cur, char *attr)
- int **IO::getIntAttr** (xmlDocPtr doc, char *tag, char *keyname, char *key, char *attr)
- int **IO::parseWord** (xmlDocPtr doc, xmlNodePtr cur, char *keyword, char *result)
- int **IO::parseInt** (xmlDocPtr doc, xmlNodePtr cur, char *keyword)
- int **IO::parseFloat** (xmlDocPtr doc, xmlNodePtr cur, char *keyword, REAL &result)
- double **IO::parseFloat** (xmlDocPtr doc, xmlNodePtr cur, char *keyword)
- void **IO::getTimeXML** (char inpfilename[])
- void **IO::getTime** (char inpfilename[])
- int **IO::getIter** (char inpfilename[])

Variables

- int **IO::ioutput** = 0
- int **IO::nxwdump** = 0

7.12 io.h File Reference

Namespaces

- namespace **IO**

Functions

- int **IO::getCharAttr** (xmlNodePtr cur, char *attr, char *result)
- int **IO::getIntAttr** (xmlNodePtr cur, char *attr)
- int **IO::getIntAttr** (xmlDocPtr doc, char *tag, char *keyname, char *key, char *attr)
- int **IO::parseWord** (xmlDocPtr doc, xmlNodePtr cur, char *keyword, char *result)
- int **IO::parseInt** (xmlDocPtr doc, xmlNodePtr cur, char *keyword)
- double **IO::parseFloat** (xmlDocPtr doc, xmlNodePtr cur, char *keyword)
- int **IO::parseFloat** (xmlDocPtr doc, xmlNodePtr cur, char *keyword, REAL &result)
- int **IO::getIter** (char inpfilename[])
- void **IO::getTime** (char inpfilename[])
- void **IO::xwd** (char *windowname)

Variables

- int **IO::ioutput**

7.13 job.cc File Reference

The ReMoDy backend.

```
#include <iostream>
#include <libxml/xmlmemory.h>
#include <libxml/parser.h>
#include <math.h>
#include "def.h"
#include "io.h"
#include "run.h"
#include "list.h"
#include "collection.h"
#include "model.h"
#include "species.h"
#include "domain.h"
```

Functions

- void **usage** ()
- void **parsename** (char *name, char *filename)
- int **main** (int argc, char *argv[])

  The Main routine.

7.13.1 Detailed Description

The ReMoDy backend.

Executes background job.

Definition in file **job.cc**.

7.13.2 Function Documentation

7.13.2.1 int main (int *argc*, char * *argv*[])

The Main routine.

parses command line arguments

creates a domain instance

erases screen

runs the code

retrieves taskname

saves data

Definition at line 32 of file job.cc.

References Run::configfile, Gui::domain, ESC, IO::getIter(), Run::init(), Run::inputfile, Run::option, parsename(), Domain::run(), Domain::save(), usage(), and WORDLENGTH.

7.13.2.2 void parsename (char * *name*, char * *filename*)

strips the suffix from the filename

Definition at line 24 of file job.cc.

Referenced by main().

7.13.2.3 void usage ()

Definition at line 21 of file job.cc.

References Run::programname.

Referenced by main().

7.14 list.h File Reference

Classes

- struct **Pointer< Element >**
- class **List< Element >**

7.15 model.cc File Reference

```
#include <math.h>
#include <iostream>
#include "def.h"
#include "run.h"
#include "list.h"
#include "collection.h"
#include "model.h"
#include "species.h"
#include "domain.h"
```

Namespaces

- namespace **Potential**

Functions

- REAL **Potential::invdist** (REAL x)
- REAL **Geom::normalize** (REAL a[])
- void **Geom::distvec** (REAL a[], REAL b[], REAL c[])
- REAL **Geom::distance** (REAL a[], REAL b[])
- REAL **Geom::distance** (int dim, REAL a[], REAL b[])
- REAL **Geom::sclp** (REAL a[], REAL b[])
- int **Geom::hash** (int **i**, int j, int **n**)
- REAL **Geom::length** (REAL a[])
- REAL **Geom::area** (REAL a[], REAL b[])

Variables

- REAL **Potential::sigma** = 1.0
- REAL **Potential::eta** = 1.0
- REAL **Potential::cutoff** = 2.0

7.16 model.h File Reference

Namespaces

- namespace **Potential**
- namespace **Geom**

Classes

- class **Molecule**

Defines

- #define **HARDBALLS**
  hardball collision as opposed to interaction via a potential

Enumerations

- enum **AtomType** {
  undefined = 0, **hydrogen** = 1, **helium** = 2, **carbon** = 12,
  oxygen = 16, **maxAtomTypes** }

Functions

- void **Potential::strength** (REAL strength)
- REAL **Potential::strength** ()
- void **Potential::lengthscale** (REAL lengthscale)
- REAL **Potential::lengthscale** ()
- void **Potential::Cutoff** (REAL newcutoff)
- REAL **Potential::Cutoff** ()
- REAL **Potential::value** (REAL r)
- REAL **Potential::force** (REAL r)
- REAL **Potential::derivative** (REAL r)
- void **Geom::distvec** (REAL a[], REAL b[], REAL c[])
- REAL **Geom::distance** (REAL a[], REAL b[])
- REAL **Geom::distance** (int dim, REAL a[], REAL b[])
- REAL **Geom::sclp** (REAL a[], REAL b[])
- REAL **Geom::length** (REAL a[])
- REAL **Geom::area** (REAL a[], REAL b[])
- REAL **Geom::normalize** (REAL a[])
- int **Geom::hash** (int **i**, int j, int **n**)

Variables

- const REAL **Potential::small** = 1.0e-20
- const REAL **Potential::large** = 1.0e20
- REAL **Potential::cutoff**
- REAL **Potential::sigma**
- REAL **Potential::eta**

7.16.1 Define Documentation

7.16.1.1 #define HARDBALLS

hardball collision as opposed to interaction via a potential

Definition at line 2 of file model.h.

7.16.2 Enumeration Type Documentation

7.16.2.1 enum AtomType

Enumerator:

undefined
hydrogen
helium
carbon
oxygen
maxAtomTypes

Definition at line 9 of file model.h.

7.17 molecules.m File Reference

Functions

- if **X** (**i**)< 0.0 Y(**i**)

Variables

- http **__pad0__**
- http **n**
- for **i**
- end end plot(X, Y) Y=0.0 for **i**
- else **y** = 1.0-2.0*(Y(**i**)-0.5)

7.17.1 Function Documentation

7.17.1.1 if X (i)

7.17.2 Variable Documentation

7.17.2.1 http __pad0__

Definition at line 76 of file molecules.m.

7.17.2.2 end end plot (X,Y) Y = 0.0 for i

Initial value:

```
1:n
        if Y(i)<=0.5
                y=2*Y(i)
```

Definition at line 87 of file molecules.m.

7.17.2.3 for i

Initial value:

```
1:n
        Y(i)=1.0-exp(-abs(X(i)))
```

Definition at line 77 of file molecules.m.

Referenced by Domain::boundary(), Domain::computeBounds(), Molecule::Coordinates(), Molecule::copy(), Gui::display(), Geom::distance(), Geom::distvec(), Domain::Domain(), Gui::getXLimits(), GridContainer::index(), List< Element >::init(), GridContainer::init(), Collection< Element >::init(), Gui::initdisp(), Boundary::Inject(), Domain::interact(), Domain::interaction(), Molecule::KineticEnergy(), Geom::length(), Gui::Materials(), Molecule::Move(), Geom::normalize(), Pool< Element >::Pool(), GridContainer::put(), Run::readcmdline(), Gui::readconf(), Gui::readparam(), Domain::run(), Geom::sclp(), Molecule::setCoordinates(), Molecule::setVelocity(), Species::Specie::Specie(), Run::testrnd(), and Molecule::Velocity().

7.17.2.4 http n

Definition at line 76 of file molecules.m.

Referenced by Pool< Element >::check(), GridContainer::index(), GridContainer::init(), Gui::query_extension(), and Gui::runmany().

7.17.2.5 else y = 1.0-2.0∗(Y(i)-0.5)

Definition at line 92 of file molecules.m.

Referenced by Domain::boundary().

7.18 remody.dox File Reference

7.19 run.cc File Reference

```
#include <stdlib.h>
#include <iostream>
#include <sys/timeb.h>
#include <libxml/xmlmemory.h>
#include <libxml/parser.h>
#include "def.h"
#include "run.h"
#include "io.h"
```

Namespaces

- namespace **Run**

Functions

- void **Run::testrnd** ()
- REAL **Run::rnd** ()
- void **Run::init** (int argc, char *argv[])
- void **Run::readcmdline** (int argc, char *argv[])

Variables

- **Option Run::option** = {0,0,0,0}
- void(*) **Run::usage** ()
- char **Run::programname** [MAXLINLEN]
- char **Run::configfile** [MAXLINLEN]
- char **Run::outputfile** [MAXLINLEN]
- char **Run::outputname** [MAXLINLEN]
- char **Run::inputfile** [MAXLINLEN]
- Time **Run::time**

7.20 run.h File Reference

Namespaces

- namespace **Run**

Classes

- struct **Option**
- struct **Run::Time**

Functions

- void **Run::init** (int argc, char *argv[])
- void **Run::readcmdline** (int argc, char *argv[])
- REAL **Run::rnd** ()

Variables

- **Option Run::option**
- Time **Run::time**
- void(*) **Run::usage** ()

7.21 species.cc File Reference

```
#include <iostream>
#include "def.h"
#include "species.h"
```

Namespaces

- namespace **Species**

Variables

- int **Species::nspecies**
- Specie * **Species::species**
- Reaction * **Species::reactions**

7.22 species.h File Reference

Namespaces

- namespace **Species**

Classes

- class **Species::Specie**
- struct **Species::Gas**

  Gas *(p. 77) is a specie with some additional parameters.*

- struct **Species::Reaction**

  Reaction *(p. 98) determines the two products only.*

- struct **Species::Reaction::Outcome**

  Possible outcomes of a reaction.

Enumerations

- enum **Species::Interaction** { **Species::missed** = 0, **Species::collided**, **Species::reacted**, **Species::annihalated** }

Variables

- int **Species::nspecies**
- Specie * **Species::species**
- Reaction * **Species::reactions**

7.23 view.cc File Reference

The frontend of ReMoDy with the OpenGL Visualizer.

```
#include <iostream>
#include <libxml/xmlmemory.h>
#include <libxml/parser.h>
#include <math.h>
#include "def.h"
#include "run.h"
#include "list.h"
#include "collection.h"
#include "grid.h"
#include "container.h"
#include "model.h"
#include "species.h"
#include "domain.h"
#include "gui.h"
```

Functions

- void **usage** ()
- int **main** (int argc, char *argv[])

  The main routine.

7.23.1 Detailed Description

The frontend of ReMoDy with the OpenGL Visualizer.

Executes a job and displays the computational domain in an OpenGL window.

Definition in file **view.cc**.

7.23.2 Function Documentation

7.23.2.1 int main (int *argc*, char * *argv*[])

The main routine.

parses command line arguments

creates a domain instance

initializes the domain

executes the job

Definition at line 29 of file view.cc.

References Gui::domain, Gui::init(), Run::init(), Run::inputfile, Gui::run(), and usage().

7.23.2.2 void usage ()

Definition at line 25 of file view.cc.

References Run::programname.