Langmuir

Generated by Doxygen 1.8.8

Fri Dec 5 2014 14:57:16

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 color Namespace Reference

Functions

- float * qColorToArray4 (const QColor &color, float *array)

  Copy color data to array of size 4.
- float * qColorToArray4 (const QColor &color)

  Copy color data to array of size 4 (static)

5.1.1 Function Documentation

5.1.1.1 float* color::qColorToArray4 (const QColor & *color,* float * *array*)

Copy color data to array of size 4.

5.1.1.2 float* color::qColorToArray4 (const QColor & *color*)

Copy color data to array of size 4 (static)

5.2 Langmuir Namespace Reference

Classes

- class Agent

  A class that abstractly represents an object that can occupy grid sites.
- class Box
- class Button
- class CarrierWriter

  A class to output carrier stats (lifetime and pathlength)
- class ChargeAgent

  A class to represent moving charged particles.
- class CheckBox
- class CheckPointer

  A class to read and write checkpoint files.
- class ColoredObject
- class CommandLineParser

A class to parse command line arguments.

- struct ConfigurationInfo

  A struct to temporarily store site IDs.
- class Controls
- class DrainAgent

  A class to remove charges.
- class DSpinBox
- class ElectronAgent

  A class to represent moving negative charges.
- class ElectronDrainAgent

  A class to remove ElectronAgents.
- class ElectronSourceAgent

  A class to inject ElectronAgents.
- class ExcitonSourceAgent

  A class to inject Excitons.
- class ExcitonWriter

  A class to output exciton stats (lifetime and pathlength)
- class FluxAgent

  A class to change the number of carriers in the system.
- class FluxWriter

  A class to output source and drain info.
- class Grid

  A class to hold Agents, calculate their positions, and store the background potential.
- class GridImage

  A class to draw images of the grid.
- class GridViewGL
- class HoleAgent

  A class to represent moving positive charges.
- class HoleDrainAgent

  A class to remove HoleAgents.
- class HoleSourceAgent

  A class to inject HoleAgents.
- class KeyValueParser

  A class to read the parameters and store them in the correct place.
- class Logger

  A class that organizes output.
- class MainWindow
- class Navigator
- class NodeFileParser
- class OpenClHelper

  A Class to run OpenCL calculations.
- class OutputInfo

  A class to generate file names using the SimulationParameters.
- class OutputStream

  A class to combine QFile, QTextStream and OutputInfo (QFileInfo).
- class PointArray
- class Potential

  A class to calculate the potential.
- class Random

  A class to generate random numbers.
- class RecombinationAgent

A class to remove Excitons.

- class RecordDialog
- class SceneOptions
- class Simulation

  A class to orchestrate the calculation.
- struct SimulationParameters

  A struct to store all simulation options To add new variables, follow these steps:
- class SourceAgent

  A class to inject charges.
- class SSpinBox
- class TypedVariable

  A template class to map between variable names (keys) and locations (references)
- class Variable

  A class to map between variable names (keys) and locations (references)
- class World

  A class to hold all objects in a simulation.
- class XYZWriter

  A class to output xyz files.

Functions

- QTextStream & operator<< (QTextStream &stream, const Agent::Type e)

  Output Agent type enum to stream.
- QDebug operator<< (QDebug dbg, const Agent::Type e)

  Output Agent type enum to debug information.
- static std::ostream & operator<< (std::ostream &stream, QString &string)
- static std::istream & operator>> (std::istream &stream, QString &string)
- QTextStream & operator<< (QTextStream &stream, const Grid::CubeFace e)

  Overload QTextStream for the Grid::CubeFace Enum.
- QDebug operator<< (QDebug dbg, const Grid::CubeFace e)

  Overload QDebug for the Grid::CubeFace Enum.
- void backupFile (const QString &name)

  Back up a file.
- void setCalculatedValues (SimulationParameters &par)

  sets parameters that depend upon other parameters
- void checkSimulationParameters (SimulationParameters &par)

  check the parameters, making sure they are valid
- QTextStream & operator<< (QTextStream &stream, const QDateTime &datetime)

  output QDateTime as qint64 mSecsSinceEpoch
- QTextStream & operator<< (QTextStream &stream, const Variable &variable)

  overload operator to write keyValue() to a stream
- QDebug operator<< (QDebug dbg, const Variable &variable)

  overload operator to write keyValue() to a QDebug
- std::ostream & operator<< (std::ostream &stream, Variable &variable)

  Operator overload to output to output 'key = value' to std::ostream.

5.2.1 Function Documentation

5.2.1.1 void Langmuir::backupFile (const QString & *name*)

Back up a file.

Back up the file using the current time and a revision number. The file is backed up as path/file.date.num, where num is determined by examing existing files in path with a similiar form (path/file.current_date.a_number). The file is renamed, not copied.

Parameters

name	a relative or absolute file name path

Warning

gives an error if a directory is passed instead of a file
gives an error if the file can not be renamed

5.2.1.2 void Langmuir::checkSimulationParameters (SimulationParameters & *par*) [inline]

check the parameters, making sure they are valid

5.2.1.3 QTextStream & Langmuir::operator<< (QTextStream & *stream,* const QDateTime & *datetime*) [inline]

output QDateTime as qint64 mSecsSinceEpoch

5.2.1.4 QTextStream& Langmuir::operator<< (QTextStream & *stream,* const Agent::Type *e*) [inline]

Output Agent type enum to stream.

5.2.1.5 QDebug Langmuir::operator<< (QDebug *dbg,* const Agent::Type *e*) [inline]

Output Agent type enum to debug information.

5.2.1.6 static std::ostream& Langmuir::operator<< (std::ostream & *stream,* QString & *string*) [inline], [static]

5.2.1.7 QTextStream& Langmuir::operator<< (QTextStream & *stream,* const Variable & *variable*) [inline]

overload operator to write keyValue() to a stream

Operator overload to output 'key = value' to QTextStream.

5.2.1.8 QDebug Langmuir::operator<< (QDebug *dbg,* const Variable & *variable*) [inline]

overload operator to write keyValue() to a QDebug

Operator overload to output 'key = value' to QDebug.

5.2.1.9 std::ostream& Langmuir::operator<< (std::ostream & *stream,* Variable & *variable*) [inline]

Operator overload to output to output 'key = value' to std::ostream.

Operator overload to output to output 'key = value' to std::ofstream.

5.2.1.10 QTextStream& Langmuir::operator<< (QTextStream & *stream,* const Grid::CubeFace *e*)

Overload QTextStream for the Grid::CubeFace Enum.

5.2.1.11 QDebug Langmuir::operator<< (QDebug *dbg,* const Grid::CubeFace *e*)

Overload QDebug for the Grid::CubeFace Enum.

5.2.1.12 static std::istream& Langmuir::operator>> (std::istream & *stream,* QString & *string*) [inline], [static]

5.2.1.13 void Langmuir::setCalculatedValues (SimulationParameters & *par*) [inline]

sets parameters that depend upon other parameters

5.3 MarchingCubes Namespace Reference

Classes

- class Isosurface

  A class to compute a contour iso-surface.
- class Triangle

  Container for vertices and normals of triangle.

Typedefs

- typedef boost::multi_array
  < float, 3 > scalar_field

Variables

- static const float a2fVertexOffset [8][3]
- static const int a2iEdgeConnection [12][2]
- static const float a2fEdgeDirection [12][3]
- static const int aiCubeEdgeFlags [256]
- static const int a2iTriangleConnectionTable [256][16]

5.3.1 Typedef Documentation

5.3.1.1 typedef boost::multi_array<float, 3> MarchingCubes::scalar_field

5.3.2 Variable Documentation

5.3.2.1 const float MarchingCubes::a2fEdgeDirection[12][3] [static]

Initial value:

```
=
    {
            {1.0, 0.0, 0.0},{0.0, 1.0, 0.0},{-1.0, 0.0, 0.0},{0.0, -1.0, 0.0},
            {1.0, 0.0, 0.0},{0.0, 1.0, 0.0},{-1.0, 0.0, 0.0},{0.0, -1.0, 0.0},
            {0.0, 0.0, 1.0},{0.0, 0.0, 1.0},{ 0.0, 0.0, 1.0},{0.0,  0.0, 1.0}
    }
```

5.3.2.2 const float MarchingCubes::a2fVertexOffset[8][3] [static]

Initial value:

```
=
    {
            {0.0, 0.0, 0.0}, {1.0, 0.0, 0.0}, {1.0, 1.0, 0.0}, {0.0, 1.0, 0.0},
            {0.0, 0.0, 1.0}, {1.0, 0.0, 1.0}, {1.0, 1.0, 1.0}, {0.0, 1.0, 1.0}
    }
```

5.3.2.3 const int MarchingCubes::a2iEdgeConnection[12][2] [static]

Initial value:

```
=
    {
            {0,1}, {1,2}, {2,3}, {3,0},
            {4,5}, {5,6}, {6,7}, {7,4},
            {0,4}, {1,5}, {2,6}, {3,7}
    }
```

5.3.2.4 const int MarchingCubes::a2iTriangleConnectionTable[256][16] [static]

5.3.2.5 const int MarchingCubes::aiCubeEdgeFlags[256] [static]

Initial value:

```
=
    {
          0x000, 0x109, 0x203, 0x30a, 0x406, 0x50f, 0x605, 0x70c, 0x80c, 0x905, 0xa0f, 0xb06, 0xc0a,
      0xd03, 0xe09, 0xf00,
          0x190, 0x099, 0x393, 0x29a, 0x596, 0x49f, 0x795, 0x69c, 0x99c, 0x895, 0xb9f, 0xa96, 0xd9a,
      0xc93, 0xf99, 0xe90,
          0x230, 0x339, 0x033, 0x13a, 0x636, 0x73f, 0x435, 0x53c, 0xa3c, 0xb35, 0x83f, 0x936, 0xe3a,
      0xf33, 0xc39, 0xd30,
          0x3a0, 0x2a9, 0x1a3, 0x0aa, 0x7a6, 0x6af, 0x5a5, 0x4ac, 0xbac, 0xaa5, 0x9af, 0x8a6, 0xfaa,
      0xea3, 0xda9, 0xca0,
          0x460, 0x569, 0x663, 0x76a, 0x066, 0x16f, 0x265, 0x36c, 0xc6c, 0xd65, 0xe6f, 0xf66, 0x86a,
      0x963, 0xa69, 0xb60,
          0x5f0, 0x4f9, 0x7f3, 0x6fa, 0x1f6, 0x0ff, 0x3f5, 0x2fc, 0xdfc, 0xcf5, 0xfff, 0xef6, 0x9fa,
      0x8f3, 0xbf9, 0xaf0,
          0x650, 0x759, 0x453, 0x55a, 0x256, 0x35f, 0x055, 0x15c, 0xe5c, 0xf55, 0xc5f, 0xd56, 0xa5a,
      0xb53, 0x859, 0x950,
          0x7c0, 0x6c9, 0x5c3, 0x4ca, 0x3c6, 0x2cf, 0x1c5, 0x0cc, 0xfcc, 0xec5, 0xdcf, 0xcc6, 0xbca,
      0xac3, 0x9c9, 0x8c0,
          0x8c0, 0x9c9, 0xac3, 0xbca, 0xcc6, 0xdcf, 0xec5, 0xfcc, 0x0cc, 0x1c5, 0x2cf, 0x3c6, 0x4ca,
      0x5c3, 0x6c9, 0x7c0,
          0x950, 0x859, 0xb53, 0xa5a, 0xd56, 0xc5f, 0xf55, 0xe5c, 0x15c, 0x055, 0x35f, 0x256, 0x55a,
      0x453, 0x759, 0x650,
          0xaf0, 0xbf9, 0x8f3, 0x9fa, 0xef6, 0xfff, 0xcf5, 0xdfc, 0x2fc, 0x3f5, 0x0ff, 0x1f6, 0x6fa,
      0x7f3, 0x4f9, 0x5f0,
          0xb60, 0xa69, 0x963, 0x86a, 0xf66, 0xe6f, 0xd65, 0xc6c, 0x36c, 0x265, 0x16f, 0x066, 0x76a,
      0x663, 0x569, 0x460,
          0xca0, 0xda9, 0xea3, 0xfaa, 0x8a6, 0x9af, 0xaa5, 0xbac, 0x4ac, 0x5a5, 0x6af, 0x7a6, 0x0aa,
      0x1a3, 0x2a9, 0x3a0,
          0xd30, 0xc39, 0xf33, 0xe3a, 0x936, 0x83f, 0xb35, 0xa3c, 0x53c, 0x435, 0x73f, 0x636, 0x13a,
      0x033, 0x339, 0x230,
          0xe90, 0xf99, 0xc93, 0xd9a, 0xa96, 0xb9f, 0x895, 0x99c, 0x69c, 0x795, 0x49f, 0x596, 0x29a,
      0x393, 0x099, 0x190,
          0xf00, 0xe09, 0xd03, 0xc0a, 0xb06, 0xa0f, 0x905, 0x80c, 0x70c, 0x605, 0x50f, 0x406, 0x30a,
      0x203, 0x109, 0x000
    }
```

5.4 Ui Namespace Reference

Chapter 6

Class Documentation

6.1 Langmuir::Agent Class Reference

A class that abstractly represents an object that can occupy grid sites.

`#include <agent.h>`

Public Types

- enum Type {
  Empty = 0, Electron = 1, Hole = 2, Defect = 3,
  Source = 4, Drain = 5, SIZE = 6 }

  An identifier for the type of Agent.

Public Member Functions

- Agent (Type type, World &world, int site=0, QObject *parent=0)

  Create an Agent.
- virtual ~Agent ()

  Destroy Agent.
- const QVector< int > & getNeighbors () const

  Get Agent neighbor list.
- void setNeighbors (QVector< int > neighbors)

  Set Agent neighbor list.
- int getCurrentSite () const

  Get Agent current site.
- int getFutureSite () const

  Get Agent future site.
- void setCurrentSite (int site)

  Set Agent current site.
- void setFutureSite (int site)

  Set Agent future site.
- Type getType () const

  Get Agent::Type enum.
- World & getWorld () const

  Get Langmuir::World reference.

Static Public Member Functions

- static QString toQString (const Agent::Type e)

  Convert Agent type enum to QString.

Protected Attributes

- int m_site

  Current site the Agent occupies.
- int m_fSite

  *Future site the Agent **will** occupy.*
- World & m_world

  Reference to World object.
- QVector< int > m_neighbors

  List fo neighboring site ids.
- Type m_type

  Agent Type enum.

6.1.1 Detailed Description

A class that abstractly represents an object that can occupy grid sites.

Agents can be Electrons, Holes, Defects, Sources, or Drains. The Agent class encodes basic information that all agents have, regardless of their type. For examples, all agents occupy a grid site, have knowledge of their neighboring sites, and know their own type.

6.1.2 Member Enumeration Documentation

6.1.2.1 enum Langmuir::Agent::Type

An identifier for the type of Agent.

Enumerator

Empty Empty Grid site.

Electron ElectronAgent.

Hole HoleAgent.

Defect Defective Grid site.

Source SourceAgent.

Drain DrainAgent.

SIZE Number of Agent Types.

6.1.3 Constructor & Destructor Documentation

6.1.3.1 Langmuir::Agent::Agent (Type *type*, World & *world*, int *site* = 0, QObject * *parent* = 0) [inline]

Create an Agent.

Parameters

type	identifier enum • for example: Electron, Hole, etc.
world	reference world object
site	grid site id Agent occupies
parent	parent QObject

6.1.3.2 Langmuir::Agent::~Agent ( ) [inline], [virtual]

Destroy Agent.

6.1.4 Member Function Documentation

6.1.4.1 int Langmuir::Agent::getCurrentSite ( ) const [inline]

Get Agent current site.

6.1.4.2 int Langmuir::Agent::getFutureSite ( ) const [inline]

Get Agent future site.

6.1.4.3 const QVector< int > & Langmuir::Agent::getNeighbors ( ) const [inline]

Get Agent neighbor list.

6.1.4.4 Agent::Type Langmuir::Agent::getType ( ) const [inline]

Get Agent::Type enum.

6.1.4.5 World & Langmuir::Agent::getWorld ( ) const [inline]

Get Langmuir::World reference.

6.1.4.6 void Langmuir::Agent::setCurrentSite (int *site*) [inline]

Set Agent current site.

6.1.4.7 void Langmuir::Agent::setFutureSite (int *site*) [inline]

Set Agent future site.

6.1.4.8 void Langmuir::Agent::setNeighbors (QVector< int > *neighbors*) [inline]

Set Agent neighbor list.

6.1.4.9 QString Langmuir::Agent::toQString (const Agent::Type *e*) [inline], [static]

Convert Agent type enum to QString.

6.1.5 Member Data Documentation

6.1.5.1 int Langmuir::Agent::m_fSite [protected]

Future site the Agent **will** occupy.

6.1.5.2 QVector<int> Langmuir::Agent::m_neighbors [protected]

List fo neighboring site ids.

6.1.5.3 int Langmuir::Agent::m_site [protected]

Current site the Agent occupies.

6.1.5.4 Type Langmuir::Agent::m_type [protected]

Agent Type enum.

6.1.5.5 World& Langmuir::Agent::m_world [protected]

Reference to World object.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/agent.h

6.2 Axis Class Reference

A class to represent an xyz axis.

`#include <axis.h>`

Public Slots

- void setXColor (QColor color)

  set the color of the x-axis
- void setYColor (QColor color)

  set the color of the x-axis
- void setZColor (QColor color)

  set the color of the x-axis
- void setRadius (double value)

  set the radius of axes
- void setLength (double value)

  set the length of axes
- virtual void makeConnections ()

  make signal/slot connections

Signals

- void xColorChanged (QColor color)
  signal that the color of x-axis has changed
- void yColorChanged (QColor color)
  signal that the color of y-axis has changed
- void zColorChanged (QColor color)
  signal that the color of z-axis has changed
- void radiusChanged (double value)
  signal that the axes radius has changed
- void lengthChanged (double value)
  signal that the axes length has changed

Public Member Functions

- Axis (LangmuirViewer &viewer, QObject *parent=0)
  create the Axis
- const QColor & getXColor () const
  get color of x-axis
- const QColor & getYColor () const
  get color of y-axis
- const QColor & getZColor () const
  get color of z-axis
- double getLength () const
  get axis length
- double getRadius () const
  get axis radius

Protected Member Functions

- virtual void init ()
  initialize object
- virtual void draw ()
  perform OpenGL drawing operations

Protected Attributes

- double m_radius
  axes radius
- double m_length
  axes length
- QColor m_xcolor
  color of x-axis
- QColor m_ycolor
  color of y-axis
- QColor m_zcolor
  color of z-axis

6.2.1 Detailed Description

A class to represent an xyz axis.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 Axis::Axis (LangmuirViewer & *viewer*, QObject * *parent* = 0) [explicit]

create the Axis

Parameters

viewer	the viewer
parent	QObject this belongs to

6.2.3 Member Function Documentation

6.2.3.1 virtual void Axis::draw ( ) [protected], [virtual]

perform OpenGL drawing operations

Reimplemented from SceneObject.

6.2.3.2 double Axis::getLength ( ) const

get axis length

6.2.3.3 double Axis::getRadius ( ) const

get axis radius

6.2.3.4 const QColor& Axis::getXColor ( ) const

get color of x-axis

6.2.3.5 const QColor& Axis::getYColor ( ) const

get color of y-axis

6.2.3.6 const QColor& Axis::getZColor ( ) const

get color of z-axis

6.2.3.7 virtual void Axis::init ( ) [protected], [virtual]

initialize object

Reimplemented from SceneObject.

Reimplemented in CornerAxis.

6.2.3.8 void Axis::lengthChanged (double *value*) [signal]

signal that the axes length has changed

Parameters

value	value of length

6.2.3.9 virtual void Axis::makeConnections ( ) [virtual], [slot]

make signal/slot connections

6.2.3.10 void Axis::radiusChanged (double *value*) [signal]

signal that the axes radius has changed

Parameters

value	value of radius

6.2.3.11 void Axis::setLength (double *value*) [slot]

set the length of axes

Parameters

value	length to set

6.2.3.12 void Axis::setRadius (double *value*) [slot]

set the radius of axes

Parameters

value	radius to set

6.2.3.13 void Axis::setXColor (QColor *color*) [slot]

set the color of the x-axis

Parameters

color	color to set

6.2.3.14 void Axis::setYColor (QColor *color*) [slot]

set the color of the x-axis

Parameters

color	color to set

6.2.3.15 void Axis::setZColor (QColor *color*) [slot]

set the color of the x-axis

Parameters

color	color to set

6.2.3.16 void Axis::xColorChanged (QColor *color*) [signal]

signal that the color of x-axis has changed

Parameters

color	value of color

6.2.3.17 void Axis::yColorChanged (QColor *color*) [signal]

signal that the color of y-axis has changed

Parameters

color	value of color

6.2.3.18 void Axis::zColorChanged (QColor *color*) [signal]

signal that the color of z-axis has changed

Parameters

color	value of color

6.2.4 Member Data Documentation

6.2.4.1 double Axis::m_length [protected]

axes length

6.2.4.2 double Axis::m_radius [protected]

axes radius

6.2.4.3 QColor Axis::m_xcolor [protected]

color of x-axis

6.2.4.4 QColor Axis::m_ycolor [protected]

color of y-axis

6.2.4.5 QColor Axis::m_zcolor [protected]

color of z-axis

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

- /home/adam/opt/langmuir/src/langmuirView/include/axis.h

6.3 Box Class Reference

A class to represent a textured box.

`#include <box.h>`

Public Types

- enum Face {
  None = 1, North = 2, South = 4, East = 8,
  West = 16, Front = 32, Back = 64, All = North | South | East | West | Front | Back }

  the faces to show texture on

Public Slots

- virtual void makeConnections ()

  make signal/slot connections
- void setColor (QColor color)

  set the color
- void setSize (double xvalue, double yvalue, double zvalue, unsigned int tesselate_x=10, unsigned int tesselate_y=10, unsigned int tesselate_z=10)

  set the box size
- void setFaces (Faces faces)

  set the list of faces to show texture on
- void showImage (bool on=true)

  show the texture
- void toggleImage ()

  toggle if texture is shown
- void setTexture (GLuint imageID)

  set texture

Signals

- void colorChanged (QColor color)

  signal that the color of has changed
- void sizeChanged (double xvalue, double yvalue, double zvalue)

  signal that the box size has changed
- void imageOnChanged (bool drawn)

  signal that the texture is drawn changed
- void facesChanged (Faces faces)

  signal that which faces the texture appears on changed
- void textureChanged (GLuint imageID)

  signal that the texture has changed

Public Member Functions

- Box (LangmuirViewer &viewer, QObject *parent=0)

  create the Box
- ~Box ()

  destroy the Box
- const QColor & getColor () const

get color

- int getXSize () const
  get x length
- int getYSize () const
  get y length
- int getZSize () const
  get z length
- bool imageIsOn () const
  true if texture is showing

Protected Member Functions

- virtual void init ()
  initialize object
- virtual void buildGeometry (unsigned int tesselate_x, unsigned int tesselate_y, unsigned int tesselate_z)
  build cube geometry
- virtual void draw ()
  perform OpenGL drawing operations

Protected Attributes

- QColor m_color
  color of box
- double m_xsize
  x length
- double m_ysize
  y length
- double m_zsize
  z length
- double m_halfXSize
  half xsize
- double m_halfYSize
  half ysize
- double m_halfZSize
  half zsize
- GLuint m_imageID
  texture ID
- bool m_imageOn
  show texture
- Faces m_faces
  texture faces
- QOpenGLBuffer * m_verticesVBO
  vertices buffer
- QOpenGLBuffer * m_normalsVBO
  normals buffer
- QOpenGLBuffer * m_texturesVBO
  texture buffer
- QOpenGLBuffer * m_indexVBO
  index buffer CW
- unsigned int m_numVertices
  *number of vertices (3 * number of points)*
- unsigned int m_numIndices
  index count

6.3.1 Detailed Description

A class to represent a textured box.

6.3.2 Member Enumeration Documentation

6.3.2.1 enum Box::Face

the faces to show texture on

Enumerator

None

North do not show texture on any face

South show texture on the +y

East show texture on the -y

West show texture on the +x

Front show texture on the -x

Back show texture on the +z

All show texture on the -z

6.3.3 Constructor & Destructor Documentation

6.3.3.1 Box::Box (LangmuirViewer & *viewer,* QObject * *parent* = 0) [explicit]

create the Box

Parameters

viewer	the viewer
parent	QObject this belongs to

6.3.3.2 Box::~Box ( )

destroy the Box

6.3.4 Member Function Documentation

6.3.4.1 virtual void Box::buildGeometry (unsigned int *tesselate_x,* unsigned int *tesselate_y,* unsigned int *tesselate_z*) [protected], [virtual]

build cube geometry

6.3.4.2 void Box::colorChanged (QColor *color*) [signal]

signal that the color of has changed

Parameters

color	value of color

6.3.4.3 virtual void Box::draw ( ) [protected], [virtual]

perform OpenGL drawing operations

Reimplemented from SceneObject.

6.3.4.4 void Box::facesChanged (Faces *faces*) [signal]

signal that which faces the texture appears on changed

Parameters

faces	list of faces

6.3.4.5 const QColor& Box::getColor ( ) const

get color

6.3.4.6 int Box::getXSize ( ) const

get x length

6.3.4.7 int Box::getYSize ( ) const

get y length

6.3.4.8 int Box::getZSize ( ) const

get z length

6.3.4.9 bool Box::imageIsOn ( ) const

true if texture is showing

6.3.4.10 void Box::imageOnChanged (bool *drawn*) [signal]

signal that the texture is drawn changed

Parameters

drawn	true if texture is drawn

6.3.4.11 virtual void Box::init ( ) [protected], [virtual]

initialize object

Reimplemented from SceneObject.

6.3.4.12 virtual void Box::makeConnections ( ) [virtual], [slot]

make signal/slot connections

6.3.4.13 void Box::setColor (QColor *color*) [slot]

set the color

Parameters

color	color to set

6.3.4.14 void Box::setFaces (Faces *faces*) [slot]

set the list of faces to show texture on

Parameters

faces	list of faces

6.3.4.15 void Box::setSize (double *xvalue*, double *yvalue*, double *zvalue*, unsigned int *tesselate_x* = 10, unsigned int *tesselate_y* = 10, unsigned int *tesselate_z* = 10) [slot]

set the box size

Parameters

xvalue	length
yvalue	width
zvalue	height

6.3.4.16 void Box::setTexture (GLuint *imageID*) [slot]

set texture

6.3.4.17 void Box::showImage (bool *on* = true) [slot]

show the texture

Parameters

on	true if texture is to be shown

6.3.4.18 void Box::sizeChanged (double *xvalue*, double *yvalue*, double *zvalue*) [signal]

signal that the box size has changed

Parameters

xvalue	value of length
yvalue	value of width

zvalue	value of height

6.3.4.19 void Box::textureChanged (GLuint *imageID*) [signal]

signal that the texture has changed

Parameters

imageID	texture ID

6.3.4.20 void Box::toggleImage ( ) [slot]

toggle if texture is shown

6.3.5 Member Data Documentation

6.3.5.1 QColor Box::m_color [protected]

color of box

6.3.5.2 Faces Box::m_faces [protected]

texture faces

6.3.5.3 double Box::m_halfXSize [protected]

half xsize

6.3.5.4 double Box::m_halfYSize [protected]

half ysize

6.3.5.5 double Box::m_halfZSize [protected]

half zsize

6.3.5.6 GLuint Box::m_imageID [protected]

texture ID

6.3.5.7 bool Box::m_imageOn [protected]

show texture

6.3.5.8 QOpenGLBuffer* Box::m_indexVBO [protected]

index buffer CW

6.3.5.9 QOpenGLBuffer* Box::m_normalsVBO [protected]

normals buffer

6.3.5.10 unsigned int Box::m_numIndices [protected]

index count

6.3.5.11 unsigned int Box::m_numVertices [protected]

number of vertices (3 * number of points)

6.3.5.12 QOpenGLBuffer* Box::m_texturesVBO [protected]

texture buffer

6.3.5.13 QOpenGLBuffer* Box::m_verticesVBO [protected]

vertices buffer

6.3.5.14 double Box::m_xsize [protected]

x length

6.3.5.15 double Box::m_ysize [protected]

y length

6.3.5.16 double Box::m_zsize [protected]

z length

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

- /home/adam/opt/langmuir/src/langmuirView/include/box.h

6.4 Langmuir::Box Class Reference

`#include <gridview.h>`

Public Slots

- void setTexture (int tid)

Public Member Functions

- Box (QObject *parent, QVector3D dimensions, QVector3D origin)
- ~Box ()
- void draw ()

Private Attributes

- QGLBuffer vBuffer
- QGLBuffer nBuffer
- QGLBuffer tBuffer
- int tid

Additional Inherited Members

6.4.1 Constructor & Destructor Documentation

6.4.1.1 Langmuir::Box::Box (QObject * *parent,* QVector3D *dimensions,* QVector3D *origin*)

6.4.1.2 Langmuir::Box::~Box ( )

6.4.2 Member Function Documentation

6.4.2.1 void Langmuir::Box::draw ( )

6.4.2.2 void Langmuir::Box::setTexture (int *tid*) [slot]

6.4.3 Member Data Documentation

6.4.3.1 QGLBuffer Langmuir::Box::nBuffer [private]

6.4.3.2 QGLBuffer Langmuir::Box::tBuffer [private]

6.4.3.3 int Langmuir::Box::tid [private]

6.4.3.4 QGLBuffer Langmuir::Box::vBuffer [private]

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.5 Langmuir::Button Class Reference

```
#include <gridview.h>
```

Public Slots

- void setTextSlot (QString value)
- void setColorSlot (QColor color)

Public Member Functions

- Button (QWidget *parent)

6.5.1 Constructor & Destructor Documentation

6.5.1.1 Langmuir::Button::Button (QWidget * *parent*) [inline]

6.5.2 Member Function Documentation

6.5.2.1 void Langmuir::Button::setColorSlot (QColor *color*) [slot]

6.5.2.2 void Langmuir::Button::setTextSlot (QString *value*) [slot]

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.6 Langmuir::CarrierWriter Class Reference

A class to output carrier stats (lifetime and pathlength)

`#include <writer.h>`

Public Member Functions

- CarrierWriter (World &world, const QString &name, QObject *parent=0)

  constructs the writer, has the same parameters as OutputInfo
- void write (ChargeAgent &charge)

  write the charge carrier statistics to the stream

Protected Attributes

- World & m_world

  reference to the world object
- OutputStream m_stream

  output file stream

6.6.1 Detailed Description

A class to output carrier stats (lifetime and pathlength)

6.6.2 Constructor & Destructor Documentation

6.6.2.1 Langmuir::CarrierWriter::CarrierWriter (World & *world*, const QString & *name*, QObject * *parent* = 0)

constructs the writer, has the same parameters as OutputInfo

6.6.3 Member Function Documentation

6.6.3.1 void Langmuir::CarrierWriter::write (ChargeAgent & *charge*)

write the charge carrier statistics to the stream

6.6.4 Member Data Documentation

6.6.4.1 OutputStream Langmuir::CarrierWriter::m_stream [protected]

output file stream

6.6.4.2 World& Langmuir::CarrierWriter::m_world [protected]

reference to the world object

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

- /home/adam/opt/langmuir/src/langmuirCore/include/writer.h

6.7 Langmuir::ChargeAgent Class Reference

A class to represent moving charged particles.

`#include <chargeagent.h>`

Public Member Functions

- ChargeAgent (Agent::Type getType, World &world, Grid &grid, int site, QObject *parent=0)
  Construct charge.
- virtual ~ChargeAgent ()
  Destroy charge.
- int charge ()
  Get the charge of the ChargeAgent.
- void chooseFuture ()
  Propose a random site to move to.
- void decideFuture ()
  Decide what should happen, called after chooseFuture.
- void completeTick ()
  Perform action, called after decideFuture.
- bool removed ()
  True if decideFuture removed the charge from the grid.
- int lifetime ()
  Number of steps ChargeAgent has existed.
- int pathlength ()
  Number of sites ChargeAgent has traversed.
- void setOpenCLID (int id)
  Set the ChargeAgent OpenCL identifier.
- int getOpenCLID ()
  Get the ChargeAgent OpenCL identifier.
- double coulombInteraction ()
  Perform coulombCPU() or coulombGPU()
- void coulombCPU ()
  Calculate the Coulomb potential on the CPU.
- void coulombGPU ()
  ***Retrieve** the Coulomb potential from the GPU*
- void compareCoulomb ()

compare results for CPU and GPU Coulomb (assumes kernel was called)

- Grid & getGrid ()

  Get the grid this ChargeAgent exists in.
- void setRemoved (const bool &status=true)

  Set the removed status of this ChargeAgent.
- virtual Agent::Type otherType ()=0

  Return the opposite ChargeAgent type relative to this ChargeAgent.
- virtual Grid & otherGrid ()=0

  Return the opposite Grid relative to this ChargeAgent's Agent::Type.

Protected Member Functions

- virtual double bindingPotential (int site)=0

  Calculate the exciton binding energy.

Protected Attributes

- int m_charge

  Charge of ChargeAgent (in units of e)
- bool m_removed

  Removed status of ChargeAgent.
- int m_lifetime

  Number of steps ChargeAgent as been in existance.
- int m_pathlength

  Number of grid spaces ChargeAgent has moved.
- Grid & m_grid

  The Grid the ChargeAgent lives in.
- int m_openCLID

  The index of the Charge in the OpenCL vectors (see OpenCLHelper)
- double m_de

  The difference in Coulomb potential between ChargeAgent::m_site and ChargeAgent::m_fSite.

Additional Inherited Members

6.7.1 Detailed Description

A class to represent moving charged particles.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 Langmuir::ChargeAgent::ChargeAgent (Agent::Type *getType*, World & *world*, Grid & *grid*, int *site*, QObject * *parent* = 0)

Construct charge.

ChargeAgent

Parameters

getType	Agent type; must be Agent::Electron or Agent::Hole
world	reference to world
grid	reference to grid
site	site id in grid
parent	parent QObject

6.7.2.2 virtual Langmuir::ChargeAgent::~ChargeAgent ( ) [virtual]

Destroy charge.

6.7.3 Member Function Documentation

6.7.3.1 virtual double Langmuir::ChargeAgent::bindingPotential (int *site*) [protected], [pure virtual]

Calculate the exciton binding energy.

Parameters

site	the site to check in other Grid

Returns

- $+0.5$ eV if exciton
- 0 otherwise

Implemented in Langmuir::HoleAgent, and Langmuir::ElectronAgent.

6.7.3.2 int Langmuir::ChargeAgent::charge ( )

Get the charge of the ChargeAgent.

6.7.3.3 void Langmuir::ChargeAgent::chooseFuture ( )

Propose a random site to move to.

6.7.3.4 void Langmuir::ChargeAgent::compareCoulomb ( )

compare results for CPU and GPU Coulomb (assumes kernel was called)

6.7.3.5 void Langmuir::ChargeAgent::completeTick ( )

Perform action, called after decideFuture.

6.7.3.6 void Langmuir::ChargeAgent::coulombCPU ( )

Calculate the Coulomb potential on the CPU.

Note

The result is stored in m_de

6.7.3.7 void Langmuir::ChargeAgent::coulombGPU ( )

Retrieve the Coulomb potential from the GPU

Note

The result is stored in m_de

Warning

this function assumes:

- the openCL id set for the ChargeAgent is the correct one
- the openCL kernel has been executed

6.7.3.8 double Langmuir::ChargeAgent::coulombInteraction ( )

Perform coulombCPU() or coulombGPU()

depends upon SimulationParameters::useOpenCL and SimulationParameters::okCL

Returns

ChargeAgent::m_de

6.7.3.9 void Langmuir::ChargeAgent::decideFuture ( )

Decide what should happen, called after chooseFuture.

6.7.3.10 Grid& Langmuir::ChargeAgent::getGrid ( )

Get the grid this ChargeAgent exists in.

6.7.3.11 int Langmuir::ChargeAgent::getOpenCLID ( )

Get the ChargeAgent OpenCL identifier.

See also

OpenClHelper

6.7.3.12 int Langmuir::ChargeAgent::lifetime ( )

Number of steps ChargeAgent has existed.

6.7.3.13 virtual Grid& Langmuir::ChargeAgent::otherGrid ( ) [pure virtual]

Return the opposite Grid relative to this ChargeAgent's Agent::Type.

Returns

World::holeGrid() if this chargeAgent is an Agent::Electron

Implemented in Langmuir::HoleAgent, and Langmuir::ElectronAgent.

6.7.3.14 virtual Agent::Type Langmuir::ChargeAgent::otherType ( ) [pure virtual]

Return the opposite ChargeAgent type relative to this ChargeAgent.

Returns

Agent::Hole if this ChargeAgent is an Agent::Electron

Implemented in Langmuir::HoleAgent, and Langmuir::ElectronAgent.

6.7.3.15 int Langmuir::ChargeAgent::pathlength ( )

Number of sites ChargeAgent has traversed.

6.7.3.16 bool Langmuir::ChargeAgent::removed ( )

True if decideFuture removed the charge from the grid.

6.7.3.17 void Langmuir::ChargeAgent::setOpenCLID (int *id*)

Set the ChargeAgent OpenCL identifier.

See also

OpenClHelper

6.7.3.18 void Langmuir::ChargeAgent::setRemoved (const bool & *status* = true)

Set the removed status of this ChargeAgent.

Note

Removed charges are not actually removed until completeTick() is called

6.7.4 Member Data Documentation

6.7.4.1 int Langmuir::ChargeAgent::m_charge [protected]

Charge of ChargeAgent (in units of e)

6.7.4.2 double Langmuir::ChargeAgent::m_de [protected]

The difference in Coulomb potential between ChargeAgent::m_site and ChargeAgent::m_fSite.

6.7.4.3 Grid& Langmuir::ChargeAgent::m_grid [protected]

The Grid the ChargeAgent lives in.

6.7.4.4 int Langmuir::ChargeAgent::m_lifetime [protected]

Number of steps ChargeAgent as been in existance.

6.7.4.5 int Langmuir::ChargeAgent::m_openCLID [protected]

The index of the Charge in the OpenCL vectors (see OpenCLHelper)

6.7.4.6 int Langmuir::ChargeAgent::m_pathlength [protected]

Number of grid spaces ChargeAgent has moved.

6.7.4.7 bool Langmuir::ChargeAgent::m_removed [protected]

Removed status of ChargeAgent.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/chargeagent.h

6.8 Langmuir::CheckBox Class Reference

`#include <gridview.h>`

Public Slots

- void setValueSlot (int checkState)

Public Member Functions

- CheckBox (QWidget *parent)

6.8.1 Constructor & Destructor Documentation

6.8.1.1 Langmuir::CheckBox::CheckBox (QWidget * *parent*) [inline]

6.8.2 Member Function Documentation

6.8.2.1 void Langmuir::CheckBox::setValueSlot (int *checkState*) [slot]

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.9 Langmuir::CheckPointer Class Reference

A class to read and write checkpoint files.

`#include <checkpointer.h>`

Public Types

- enum Section {
  Parameters, Electrons, Holes, Defects,
  Traps, TrapPotentials, RandomState, FluxState }

A way to identify different sections in the input file.

Public Member Functions

- CheckPointer (World &world, QObject *parent=0)

  Create the checkpointer object.
- void load (const QString &fileName, ConfigurationInfo &configInfo)

  load simulation information
- void save (const QString &fileName="%stub.chk")

  save simulation information
- void checkStream (std::istream &stream, const QString &message="")

  check to see if input stream has failed

Private Member Functions

- std::istream & loadElectrons (std::istream &stream, ConfigurationInfo &configInfo)

  load electrons sites from input file
- std::istream & loadHoles (std::istream &stream, ConfigurationInfo &configInfo)

  load hole sites from input file
- std::istream & loadDefects (std::istream &stream, ConfigurationInfo &configInfo)

  load defect sites from input file
- std::istream & loadTraps (std::istream &stream, ConfigurationInfo &configInfo)

  load trap sites from input file
- std::istream & loadTrapPotentials (std::istream &stream, ConfigurationInfo &configInfo)

  load trap energies from input file
- std::istream & loadFluxState (std::istream &stream, ConfigurationInfo &configInfo)

  load flux state from input file
- std::istream & loadParameters (std::istream &stream)

  load parameter from input file
- std::istream & loadRandomState (std::istream &stream)

  load random number generator state from input file
- std::ostream & saveElectrons (std::ostream &stream)

  save electron site ids to output file
- std::ostream & saveHoles (std::ostream &stream)

  save hole site ids to output file
- std::ostream & saveDefects (std::ostream &stream)

  save defect site ids to output file
- std::ostream & saveTraps (std::ostream &stream)

  save trap site ids to output file
- std::ostream & saveTrapPotentials (std::ostream &stream)

  save trap energies to output file
- std::ostream & saveFluxState (std::ostream &stream)

  save flux states to output file
- std::ostream & saveParameters (std::ostream &stream)

  save parameters to output file
- std::ostream & saveRandomState (std::ostream &stream)

  save random number generator state to output file

Private Attributes

- World & m_world

  reference to world object

6.9.1 Detailed Description

A class to read and write checkpoint files.

Checkpoint files are essentially the same as input files

6.9.2 Member Enumeration Documentation

6.9.2.1 enum Langmuir::CheckPointer::Section

A way to identify different sections in the input file.

Enumerator

Parameters

Electrons

Holes

Defects

Traps

TrapPotentials

RandomState

FluxState

6.9.3 Constructor & Destructor Documentation

6.9.3.1 Langmuir::CheckPointer::CheckPointer (World & *world,* QObject * *parent* = 0) [explicit]

Create the checkpointer object.

Parameters

world	reference world object
parent	parent QObject

6.9.4 Member Function Documentation

6.9.4.1 void Langmuir::CheckPointer::checkStream (std::istream & *stream,* const QString & *message* = " ")

check to see if input stream has failed

Parameters

stream	input stream
message	the error message to output if stream failed

6.9.4.2 void Langmuir::CheckPointer::load (const QString & *fileName,* ConfigurationInfo & *configInfo*)

load simulation information

Parameters

fileName	name of input file
configInfo	temporary storage for electrons, holes, etc

6.9.4.3 std::istream& Langmuir::CheckPointer::loadDefects (std::istream & *stream,* ConfigurationInfo & *configInfo*) [private]

load defect sites from input file

Parameters

stream	the input stream
configInfo	temporary storage for site ids

6.9.4.4 std::istream& Langmuir::CheckPointer::loadElectrons (std::istream & *stream,* ConfigurationInfo & *configInfo*) [private]

load electrons sites from input file

Parameters

stream	the input stream
configInfo	temporary storage for site ids

6.9.4.5 std::istream& Langmuir::CheckPointer::loadFluxState (std::istream & *stream,* ConfigurationInfo & *configInfo*) [private]

load flux state from input file

Parameters

stream	the input stream
configInfo	temporary storage for flux state

6.9.4.6 std::istream& Langmuir::CheckPointer::loadHoles (std::istream & *stream,* ConfigurationInfo & *configInfo*) [private]

load hole sites from input file

Parameters

stream	the input stream
configInfo	temporary storage for site ids

6.9.4.7 std::istream& Langmuir::CheckPointer::loadParameters (std::istream & *stream*) [private]

load parameter from input file

Parameters

stream	the input stream

6.9.4.8 std::istream& Langmuir::CheckPointer::loadRandomState (std::istream & *stream*) [private]

load random number generator state from input file

Parameters

stream	the input stream

6.9.4.9 std::istream& Langmuir::CheckPointer::loadTrapPotentials (std::istream & *stream,* ConfigurationInfo & *configInfo*) [private]

load trap energies from input file

Parameters

stream	the input stream
configInfo	temporary storage for site energies

6.9.4.10 std::istream& Langmuir::CheckPointer::loadTraps (std::istream & *stream,* ConfigurationInfo & *configInfo*) [private]

load trap sites from input file

Parameters

stream	the input stream
configInfo	temporary storage for site ids

6.9.4.11 void Langmuir::CheckPointer::save (const QString & *fileName* = "%stub.chk")

save simulation information

Parameters

fileName	name of output file

6.9.4.12 std::ostream& Langmuir::CheckPointer::saveDefects (std::ostream & *stream*) [private]

save defect site ids to output file

Parameters

stream	output stream

6.9.4.13 std::ostream& Langmuir::CheckPointer::saveElectrons (std::ostream & *stream*) [private]

save electron site ids to output file

Parameters

stream	output stream

6.9.4.14 std::ostream& Langmuir::CheckPointer::saveFluxState (std::ostream & *stream*) [private]

save flux states to output file

Parameters

stream	output stream

6.9.4.15 std::ostream& Langmuir::CheckPointer::saveHoles (std::ostream & *stream*) [private]

save hole site ids to output file

Parameters

stream	output stream

6.9.4.16 std::ostream& Langmuir::CheckPointer::saveParameters (std::ostream & *stream*) [private]

save parameters to output file

Parameters

stream	output stream

6.9.4.17 std::ostream& Langmuir::CheckPointer::saveRandomState (std::ostream & *stream*) [private]

save random number generator state to output file

Parameters

stream	output stream

6.9.4.18 std::ostream& Langmuir::CheckPointer::saveTrapPotentials (std::ostream & *stream*) [private]

save trap energies to output file

Parameters

stream	output stream

6.9.4.19 std::ostream& Langmuir::CheckPointer::saveTraps (std::ostream & *stream*) [private]

save trap site ids to output file

Parameters

stream	output stream

6.9.5 Member Data Documentation

6.9.5.1 World& Langmuir::CheckPointer::m_world [private]

reference to world object

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

- /home/adam/opt/langmuir/src/langmuirCore/include/checkpointer.h

6.10 ColorButton Class Reference

`#include <colorbutton.h>`

Public Slots

- void setButtonColor (QColor color)
- void getColor ()

Signals

- void selectedColor (QColor color)

Public Member Functions

- ColorButton (QWidget *parent=0)
- QColorDialog & colorDialog ()
- ~ColorButton ()

Protected Attributes

- QColor m_color

Static Protected Attributes

- static QColorDialog * m_colordialog

6.10.1 Constructor & Destructor Documentation

6.10.1.1 ColorButton::ColorButton (QWidget * *parent* = `0`) `[explicit]`

6.10.1.2 ColorButton::~ColorButton ( )

6.10.2 Member Function Documentation

6.10.2.1 QColorDialog& ColorButton::colorDialog ( )

6.10.2.2 void ColorButton::getColor ( ) `[slot]`

6.10.2.3 void ColorButton::selectedColor (QColor *color*) `[signal]`

6.10.2.4 void ColorButton::setButtonColor (QColor *color*) `[slot]`

6.10.3 Member Data Documentation

6.10.3.1 QColor ColorButton::m_color `[protected]`

6.10.3.2 QColorDialog* ColorButton::m_colordialog `[static]`, `[protected]`

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

- /home/adam/opt/langmuir/src/langmuirView/include/colorbutton.h

6.11 Langmuir::ColoredObject Class Reference

`#include <gridview.h>`

Public Slots

- void setColor (QColor color)
- void setInvisible (int checkState)
- void setColorDialog (QColorDialog *dialog)
- void openColorDialog ()

Signals

- void colorChanged (QColor color)

Public Member Functions

- ColoredObject (QObject *parent=0)
- const QColor & getColor () const
- const float * getLight () const

Protected Attributes

- QVector< float > light
- QColorDialog * dialog
- bool invisible
- QColor color

6.11.1 Constructor & Destructor Documentation

6.11.1.1 Langmuir::ColoredObject::ColoredObject (QObject * *parent* = `0`)

6.11.2 Member Function Documentation

6.11.2.1 void Langmuir::ColoredObject::colorChanged (QColor *color*) `[signal]`

6.11.2.2 const QColor& Langmuir::ColoredObject::getColor ( ) const

6.11.2.3 const float* Langmuir::ColoredObject::getLight ( ) const

6.11.2.4 void Langmuir::ColoredObject::openColorDialog ( ) `[slot]`

6.11.2.5 void Langmuir::ColoredObject::setColor (QColor *color*) `[slot]`

6.11.2.6 void Langmuir::ColoredObject::setColorDialog (QColorDialog * *dialog*) `[slot]`

6.11.2.7 void Langmuir::ColoredObject::setInvisible (int *checkState*) `[slot]`

6.11.3 Member Data Documentation

6.11.3.1 QColor Langmuir::ColoredObject::color `[protected]`

6.11.3.2 QColorDialog* Langmuir::ColoredObject::dialog [protected]

6.11.3.3 bool Langmuir::ColoredObject::invisible [protected]

6.11.3.4 QVector<float> Langmuir::ColoredObject::light [protected]

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.12 Langmuir::CommandLineParser Class Reference

A class to parse command line arguments.

`#include <clparser.h>`

Public Member Functions

- CommandLineParser (QObject *parent=0)
  create the CommandLineParser
- void setDescription (const QString &desc)
  set the program description
- void addBool (QString flag, QString dest, QString help)
  add a flag that has no argument
- void addPositional (QString dest, QString help)
  add an argument that has no flag
- void add (QString flag, QString dest, QString help)
  add a flag that has an argument
- template<typename T >
  T get (const QString &dest, T default_value)
  get value by key and convert to type
- void parse (QStringList &args)
  parse the command line arguments
- QString help ()
  get the help string

Protected Member Functions

- template<typename T >
  T convert (const QString &value)
  convert value to type

Protected Attributes

- QMap< QString, bool > m_isBool
  map dest->boolean
- QMap< QString, int > m_isPositional
  map dest->boolean
- QMap< QString, QString > m_flags
  map dest->flag
- QMap< QString, QString > m_helps

map dest->help

- QMap< QString, QString > m_values

  map dest->value

- QString m_description

  description string

- QStringList m_args

  list of remaining command line arguments

- unsigned int m_numPositional

  total number of positional arguments

- unsigned int m_numArguments

  total number of arguments

6.12.1 Detailed Description

A class to parse command line arguments.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 Langmuir::CommandLineParser::CommandLineParser (QObject * *parent* = 0) [explicit]

create the CommandLineParser

Parameters

parent	QObject this belongs to

6.12.3 Member Function Documentation

6.12.3.1 void Langmuir::CommandLineParser::add (QString *flag*, QString *dest*, QString *help*)

add a flag that has an argument

Parameters

flag	command line flag
dest	internal key
help	help message

6.12.3.2 void Langmuir::CommandLineParser::addBool (QString *flag*, QString *dest*, QString *help*)

add a flag that has no argument

Parameters

flag	command line flag
dest	internal key
help	help message

6.12.3.3 void Langmuir::CommandLineParser::addPositional (QString *dest*, QString *help*)

add an argument that has no flag

Parameters

dest	internal key
help	help message

6.12.3.4 template<typename T > T Langmuir::CommandLineParser::convert (const QString & *value*) [protected]

convert value to type

Parameters

value	value as string

6.12.3.5 template<typename T > T Langmuir::CommandLineParser::get (const QString & *dest*, T *default_value*)

get value by key and convert to type

Parameters

dest	internal key
default_value	default value if key not present

6.12.3.6 QString Langmuir::CommandLineParser::help ( )

get the help string

6.12.3.7 void Langmuir::CommandLineParser::parse (QStringList & *args*)

parse the command line arguments

Parameters

args	arguments as a list of strings

6.12.3.8 void Langmuir::CommandLineParser::setDescription (const QString & *desc*)

set the program description

Parameters

desc	description string

6.12.4 Member Data Documentation

6.12.4.1 QStringList Langmuir::CommandLineParser::m_args [protected]

list of *remaining* command line arguments

6.12.4.2 QString Langmuir::CommandLineParser::m_description [protected]

description string

6.12.4.3 QMap<QString,QString> Langmuir::CommandLineParser::m_flags [protected]

map dest->flag

6.12.4.4 QMap<QString,QString> Langmuir::CommandLineParser::m_helps [protected]

map dest->help

6.12.4.5 QMap<QString,bool> Langmuir::CommandLineParser::m_isBool [protected]

map dest->boolean

6.12.4.6 QMap<QString,int> Langmuir::CommandLineParser::m_isPositional [protected]

map dest->boolean

6.12.4.7 unsigned int Langmuir::CommandLineParser::m_numArguments [protected]

total number of arguments

6.12.4.8 unsigned int Langmuir::CommandLineParser::m_numPositional [protected]

total number of positional arguments

6.12.4.9 QMap<QString,QString> Langmuir::CommandLineParser::m_values [protected]

map dest->value

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

- /home/adam/opt/langmuir/src/langmuirCore/include/clparser.h

6.13 Langmuir::ConfigurationInfo Struct Reference

A struct to temporarily store site IDs.

`#include <parameters.h>`

Public Attributes

- QList< qint32 > electrons
  a list of current electron site IDs
- QList< qint32 > holes
  a list of current holes site IDs
- QList< qint32 > defects
  a list of current defects site IDs
- QList< qint32 > traps
  a list of current traps site IDs
- QList< qreal > trapPotentials
  a list of current traps site IDs
- QList< quint64 > fluxInfo
  a list of flux attempt, success values

6.13.1 Detailed Description

A struct to temporarily store site IDs.

6.13.2 Member Data Documentation

6.13.2.1 QList<qint32> Langmuir::ConfigurationInfo::defects

a list of current defects site IDs

6.13.2.2 QList<qint32> Langmuir::ConfigurationInfo::electrons

a list of current electron site IDs

6.13.2.3 QList<quint64> Langmuir::ConfigurationInfo::fluxInfo

a list of flux attempt, success values

6.13.2.4 QList<qint32> Langmuir::ConfigurationInfo::holes

a list of current holes site IDs

6.13.2.5 QList<qreal> Langmuir::ConfigurationInfo::trapPotentials

a list of current traps site IDs

6.13.2.6 QList<qint32> Langmuir::ConfigurationInfo::traps

a list of current traps site IDs

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

- /home/adam/opt/langmuir/src/langmuirCore/include/parameters.h

6.14 Langmuir::Controls Class Reference

`#include <gridview.h>`

Public Member Functions

- Controls (QWidget *parent)

Public Attributes

- QGridLayout * layout
- QList< Button * > buttons
- QList< QLabel * > labels
- QList< SSpinBox * > spinBoxes
- QList< CheckBox * > checkBoxes
- QList< QLCDNumber * > lcdNumbers

6.14.1 Constructor & Destructor Documentation

6.14.1.1 Langmuir::Controls::Controls (QWidget * *parent*)

6.14.2 Member Data Documentation

6.14.2.1 QList< Button* > Langmuir::Controls::buttons

6.14.2.2 QList< CheckBox* > Langmuir::Controls::checkBoxes

6.14.2.3 QList< QLabel* > Langmuir::Controls::labels

6.14.2.4 QGridLayout* Langmuir::Controls::layout

6.14.2.5 QList< QLCDNumber* > Langmuir::Controls::lcdNumbers

6.14.2.6 QList< SSpinBox* > Langmuir::Controls::spinBoxes

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.15 CornerAxis Class Reference

A class to represent an xyz axis that doesnt change size/position.

`#include <corneraxis.h>`

Public Types

- enum Location { LowerLeft, UpperLeft, LowerRight, UpperRight }
  the location of the axis

Public Slots

- void setLocation (Location location)
  set the axis location
- void setShift (int shift)
  set the axis shift
- void setSize (int size)
  set the axis size
- virtual void makeConnections ()
  make signal/slot connections

Signals

- void locationChanged (Location)
  signal that the axis location has changed
- void shiftChanged (int)
  signal that the axis shift has changed
- void sizeChanged (int)
  signal that the axis size has changed

Public Member Functions

- CornerAxis (LangmuirViewer &viewer, QObject *parent=0)
  create the CornerAxis
- Location getLocation () const
  get the location of the corner axis
- int getSize () const
  get the size of the viewport
- int getShift () const
  get the shift of the viewport

Protected Member Functions

- virtual void init ()
  initialize object
- virtual void preDraw ()
  perform OpenGL drawing operations before draw()
- virtual void postDraw ()
  perform OpenGL drawing operations after draw()

Private Attributes

- Location m_location
  location of axis
- int m_scissorBox [4]
  storage for scissor box
- int m_viewPort [4]
  storage for viewport
- int m_shift
  shift of axis from edge
- int m_size
  size of altered viewport

Additional Inherited Members

6.15.1 Detailed Description

A class to represent an xyz axis that doesnt change size/position.

6.15.2 Member Enumeration Documentation

6.15.2.1 enum CornerAxis::Location

the location of the axis

Enumerator

LowerLeft

UpperLeft draw axis in lower left

LowerRight draw axis in upper left

UpperRight draw axis in lower right draw axis in upper right

6.15.3 Constructor & Destructor Documentation

6.15.3.1 CornerAxis::CornerAxis (LangmuirViewer & *viewer*, QObject * *parent* = 0) [explicit]

create the CornerAxis

Parameters

viewer	the viewer
parent	QObject this belongs to

6.15.4 Member Function Documentation

6.15.4.1 Location CornerAxis::getLocation ( ) const

get the location of the corner axis

6.15.4.2 int CornerAxis::getShift ( ) const

get the shift of the viewport

6.15.4.3 int CornerAxis::getSize ( ) const

get the size of the viewport

6.15.4.4 virtual void CornerAxis::init ( ) [protected], [virtual]

initialize object

Reimplemented from Axis.

6.15.4.5 void CornerAxis::locationChanged (Location) [signal]

signal that the axis location has changed

6.15.4.6 virtual void CornerAxis::makeConnections ( ) [virtual], [slot]

make signal/slot connections

6.15.4.7 virtual void CornerAxis::postDraw ( ) [protected], [virtual]

perform OpenGL drawing operations after draw()

Reimplemented from SceneObject.

6.15.4.8 virtual void CornerAxis::preDraw ( ) [protected], [virtual]

perform OpenGL drawing operations before draw()

Reimplemented from SceneObject.

6.15.4.9 void CornerAxis::setLocation (Location *location*) [slot]

set the axis location

Parameters

location	location to set

6.15.4.10 void CornerAxis::setShift (int *shift*) [slot]

set the axis shift

Parameters

shift	shift to set

6.15.4.11 void CornerAxis::setSize (int *size*) [slot]

set the axis size

Parameters

size	size to set

6.15.4.12 void CornerAxis::shiftChanged (int) [signal]

signal that the axis shift has changed

6.15.4.13 void CornerAxis::sizeChanged (int) [signal]

signal that the axis size has changed

6.15.5 Member Data Documentation

6.15.5.1 Location CornerAxis::m_location [private]

location of axis

6.15.5.2 int CornerAxis::m_scissorBox[4] [private]

storage for scissor box

6.15.5.3 int CornerAxis::m_shift [private]

shift of axis from edge

6.15.5.4 int CornerAxis::m_size [private]

size of altered viewport

6.15.5.5 int CornerAxis::m_viewPort[4] [private]

storage for viewport

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

- /home/adam/opt/langmuir/src/langmuirView/include/corneraxis.h

6.16 Langmuir::DrainAgent Class Reference

A class to remove charges.

`#include <drainagent.h>`

Public Member Functions

- DrainAgent (World &world, Grid &grid, QObject *parent=0)
  create a DrainAgent
- virtual bool tryToAccept (ChargeAgent *charge)
  accept charge with constant probability

Additional Inherited Members

6.16.1 Detailed Description

A class to remove charges.

Unlike SourceAgents, the algorithms for removing charges currently reside in the Simulation class. This should be fixed. The DrainAgent is keeping track of drain statistics and transport probability.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 Langmuir::DrainAgent::DrainAgent (World & *world*, Grid & *grid*, QObject * *parent* = `0`)

create a DrainAgent

6.16.3 Member Function Documentation

6.16.3.1 virtual bool Langmuir::DrainAgent::tryToAccept (ChargeAgent * *charge*) `[virtual]`

accept charge with constant probability

Reimplemented in Langmuir::RecombinationAgent.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/drainagent.h

6.17 Langmuir::DSpinBox Class Reference

`#include <gridview.h>`

Public Slots

- void setValueSlot (double value)

Public Member Functions

- DSpinBox (QWidget *parent)

6.17.1 Constructor & Destructor Documentation

6.17.1.1 Langmuir::DSpinBox::DSpinBox (QWidget * *parent*) [inline]

6.17.2 Member Function Documentation

6.17.2.1 void Langmuir::DSpinBox::setValueSlot (double *value*) [slot]

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.18 Langmuir::ElectronAgent Class Reference

A class to represent moving negative charges.

`#include <chargeagent.h>`

Public Member Functions

- ElectronAgent (World &world, int site, QObject *parent=0)
  Construct ElectronAgent.

Protected Member Functions

- virtual double bindingPotential (int site)
  Calculate Exciton Binding Energy.
- virtual Agent::Type otherType ()
  Return other Agent::Type.
- virtual Grid & otherGrid ()
  Return other Grid.

Additional Inherited Members

6.18.1 Detailed Description

A class to represent moving negative charges.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 Langmuir::ElectronAgent::ElectronAgent (World & *world,* int *site,* QObject * *parent* = 0)

Construct ElectronAgent.

6.18.3 Member Function Documentation

6.18.3.1 virtual double Langmuir::ElectronAgent::bindingPotential (int *site*) [protected], [virtual]

Calculate Exciton Binding Energy.

Parameters

site	the site to check in other Grid

Returns

- $+0.5$ eV if exciton
- 0 otherwise

Implements Langmuir::ChargeAgent.

6.18.3.2 virtual Grid& Langmuir::ElectronAgent::otherGrid ( ) [protected], [virtual]

Return other Grid.

Returns

World::holeGrid

Implements Langmuir::ChargeAgent.

6.18.3.3 virtual Agent::Type Langmuir::ElectronAgent::otherType ( ) [protected], [virtual]

Return other Agent::Type.

Returns

Agent::Hole

Implements Langmuir::ChargeAgent.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/chargeagent.h

6.19 Langmuir::ElectronDrainAgent Class Reference

A class to remove ElectronAgents.

`#include <drainagent.h>`

Public Member Functions

- ElectronDrainAgent (World &world, int site, QObject *parent=0)
  create an ElectronDrainAgent at a specific site
- ElectronDrainAgent (World &world, Grid::CubeFace cubeFace, QObject *parent=0)
  create a ElectronDrainAgent at a specific Grid::CubeFace

Protected Member Functions

- virtual double energyChange (int fSite)
  same as FluxAgent::energyChange(), but specialized for ElectronAgents.

Additional Inherited Members

6.19.1 Detailed Description

A class to remove ElectronAgents.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 Langmuir::ElectronDrainAgent::ElectronDrainAgent (World & *world,* int *site,* QObject * *parent* = 0)

create an ElectronDrainAgent at a specific site

6.19.2.2 Langmuir::ElectronDrainAgent::ElectronDrainAgent (World & *world,* Grid::CubeFace *cubeFace,* QObject * *parent* = 0)

create a ElectronDrainAgent at a specific Grid::CubeFace

6.19.3 Member Function Documentation

6.19.3.1 virtual double Langmuir::ElectronDrainAgent::energyChange (int *fSite*) [protected], [virtual]

same as FluxAgent::energyChange(), but specialized for ElectronAgents.

Note really used because the default FluxAgent::shouldTransport() behavoir, which is to use a simple constant probability, has not been reimplemented for DrainAgents.

Reimplemented from Langmuir::FluxAgent.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/drainagent.h

6.20 Langmuir::ElectronSourceAgent Class Reference

A class to inject ElectronAgents.

`#include <sourceagent.h>`

Public Member Functions

- ElectronSourceAgent (World &world, int site, QObject *parent=0)
  create an ElectronSourceAgent at a specific site
- ElectronSourceAgent (World &world, Grid::CubeFace cubeFace, QObject *parent=0)
  create an ElectronSourceAgent at a specific Grid::CubeFace

Protected Member Functions

- virtual bool validToInject (int site)
  same as SourceAgent::validToInject(), but specialized for ElectronAgents.
- virtual double energyChange (int site)
  same as FluxAgent::energyChange(), but specialized for ElectronAgents.
- virtual void inject (int site)
  same as SourceAgent::inject(), but specialized for ElectronAgents.

Additional Inherited Members

6.20.1 Detailed Description

A class to inject ElectronAgents.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 Langmuir::ElectronSourceAgent::ElectronSourceAgent (World & *world,* int *site,* QObject * *parent* = 0)

create an ElectronSourceAgent at a specific site

6.20.2.2 Langmuir::ElectronSourceAgent::ElectronSourceAgent (World & *world,* Grid::CubeFace *cubeFace,* QObject * *parent* = 0)

create an ElectronSourceAgent at a specific Grid::CubeFace

6.20.3 Member Function Documentation

6.20.3.1 virtual double Langmuir::ElectronSourceAgent::energyChange (int *site*) [protected], [virtual]

same as FluxAgent::energyChange(), but specialized for ElectronAgents.

Reimplemented from Langmuir::FluxAgent.

6.20.3.2 virtual void Langmuir::ElectronSourceAgent::inject (int *site*) [protected], [virtual]

same as SourceAgent::inject(), but specialized for ElectronAgents.

Implements Langmuir::SourceAgent.

6.20.3.3 virtual bool Langmuir::ElectronSourceAgent::validToInject (int *site*) [protected], [virtual]

same as SourceAgent::validToInject(), but specialized for ElectronAgents.

Implements Langmuir::SourceAgent.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/sourceagent.h

6.21 Langmuir::ExcitonSourceAgent Class Reference

A class to inject Excitons.

`#include <sourceagent.h>`

Public Member Functions

- ExcitonSourceAgent (World &world, QObject *parent=0)

  create an ExcitonSourceAgent

Protected Member Functions

- virtual bool validToInject (int site)

  checks both grids if its ok to inject charges
- virtual double energyChange (int site)

  currently implemented as zero and not really used
- virtual bool shouldTransport (int site)

  uses the simple constant probability method
- virtual int chooseSite ()

  choose a site to inject to
- virtual void inject (int site)

  similar to SourceAgent::inject(), but injects both a HoleAgent and an ElectronAgent

Additional Inherited Members

6.21.1 Detailed Description

A class to inject Excitons.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 Langmuir::ExcitonSourceAgent::ExcitonSourceAgent (World & *world*, QObject * *parent* = 0)

create an ExcitonSourceAgent

6.21.3 Member Function Documentation

6.21.3.1 virtual int Langmuir::ExcitonSourceAgent::chooseSite ( ) [protected], [virtual]

choose a site to inject to

reimplemented to chose a site at any grid site

Reimplemented from Langmuir::SourceAgent.

6.21.3.2 virtual double Langmuir::ExcitonSourceAgent::energyChange (int *site*) [protected], [virtual]

currently implemented as zero and not really used

Reimplemented from Langmuir::FluxAgent.

6.21.3.3 virtual void Langmuir::ExcitonSourceAgent::inject (int *site*) [protected], [virtual]

similar to SourceAgent::inject(), but injects both a HoleAgent and an ElectronAgent

Implements Langmuir::SourceAgent.

6.21.3.4 virtual bool Langmuir::ExcitonSourceAgent::shouldTransport (int *site*) [protected], [virtual]

uses the simple constant probability method

Reimplemented from Langmuir::SourceAgent.

6.21.3.5 virtual bool Langmuir::ExcitonSourceAgent::validToInject (int *site*) [protected], [virtual]

checks both grids if its ok to inject charges

Implements Langmuir::SourceAgent.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/sourceagent.h

6.22 Langmuir::ExcitonWriter Class Reference

A class to output exciton stats (lifetime and pathlength)

`#include <writer.h>`

Public Member Functions

- ExcitonWriter (World &world, const QString &name, QObject *parent=0)
  constructs the writer, has the same parameters as OutputInfo
- void write (ChargeAgent &charge1, ChargeAgent &charge2, bool recombined=false)
  write the exciton statistics to the stream

Protected Attributes

- World & m_world
  reference to the world object
- OutputStream m_stream
  output file stream

6.22.1 Detailed Description

A class to output exciton stats (lifetime and pathlength)

6.22.2 Constructor & Destructor Documentation

6.22.2.1 Langmuir::ExcitonWriter::ExcitonWriter (World & *world*, const QString & *name*, QObject * *parent* = `0`)

constructs the writer, has the same parameters as OutputInfo

6.22.3 Member Function Documentation

6.22.3.1 void Langmuir::ExcitonWriter::write (ChargeAgent & *charge1*, ChargeAgent & *charge2*, bool *recombined* = `false`)

write the exciton statistics to the stream

6.22.4 Member Data Documentation

6.22.4.1 OutputStream Langmuir::ExcitonWriter::m_stream [protected]

output file stream

6.22.4.2 World& Langmuir::ExcitonWriter::m_world [protected]

reference to the world object

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

- /home/adam/opt/langmuir/src/langmuirCore/include/writer.h

6.23 Langmuir::FluxAgent Class Reference

A class to change the number of carriers in the system.

`#include <fluxagent.h>`

Public Member Functions

- FluxAgent (Agent::Type type, World &world, Grid &grid, QObject *parent=0)
  Create the flux agent.
- ~FluxAgent ()
  unregisters FluxAgent from the grid
- void setPotential (double potential)
  set the FluxAgent's potential
- double potential () const
  get the FluxAgent's potential
- void setRate (double rate)
  set the FluxAgent's rate
- void setRateSmartly (double rate, double dflt)
  set the FluxAgent's rate
- double rate () const
  get the FluxAgent's rate
- void setAttempts (unsigned long int value)
  set the FluxAgent's attempt counter
- unsigned long int attempts () const
  get the FluxAgent's attempt counter
- void setSuccesses (unsigned long int value)
  set the FluxAgent's success counter
- unsigned long int successes () const
  get the FluxAgent's success counter
- void storeLast ()
  set the value of last to the value of successes, and store the current step
- unsigned long int successesSinceLast () const
  get the number of successes since storeLast() was called
- unsigned long int attemptsSinceLast () const
  get the number of attempts since storeLast() was called
- unsigned long int stepsSinceLast () const
  get the number of steps since storeLast() was called
- double successProbability () const
  calculate and return the current probabilty of success
- double successRate () const
  calculate and return the current rate of success
- double successProbabilitySinceLast () const

calculate and return the probabilty of success since storeLast() was called

- double successRateSinceLast () const

  calculate and return the rate of success since storeLast() was called

- void resetCounters ()

  set the attempt and success counters to zero

- Grid::CubeFace face () const

  get the Grid:CubeFace this FluxAgent is assigned to

- Grid & grid () const

  get the Grid this FluxAgent belongs to

Protected Member Functions

- void initializeSite (int site)

  assign the FluxAgent to a specific site in the grid

- void initializeSite (Grid::CubeFace cubeFace)

  assign the FluxAgent to a specific Grid::CubeFace

- virtual bool shouldTransport (int site)

  decide if the FluxAgent should transport a carrier to/from a given site

- virtual double energyChange (int site)

  The energy change associated with moving a carrier from the FluxAgent to a site.

- QString faceToLetter ()

  convert the Grid::CubeFace to a single letter

Protected Attributes

- unsigned long int m_attempts

  the number of times the FluxAgent has tried to transport.

- unsigned long int m_successes

  the number of times the FluxAgent was successful in transporting.

- unsigned long int m_lastSuccesses

  storage to note the number of successes at some step

- unsigned long int m_lastAttempts

  storage to note the number of successes at some step

- unsigned long int m_lastStep

  the step at which last was noted

- double m_probability

  the constant probability used in the default behavoir of shouldTransport().

- double m_potential

  the potential that is (possibly) used when calculating an energy change

- Grid & m_grid

  the grid this FluxAgent resides in

- Grid::CubeFace m_face

  the face of the grid this FluxAgent occupies

Additional Inherited Members

6.23.1 Detailed Description

A class to change the number of carriers in the system.

A flux agent can inject carriers (Agent::Source) or accept carriers (Agent::Drain)

6.23.2 Constructor & Destructor Documentation

6.23.2.1 Langmuir::FluxAgent::FluxAgent (Agent::Type *type*, World & *world*, Grid & *grid*, QObject * *parent* = 0)

Create the flux agent.

Parameters

type	either a Agent::Source or Agent::Drain
world	reference to world object
grid	reference to grid
parent	parent QObject

6.23.2.2 Langmuir::FluxAgent::~FluxAgent ( )

unregisters FluxAgent from the grid

6.23.3 Member Function Documentation

6.23.3.1 unsigned long int Langmuir::FluxAgent::attempts ( ) const

get the FluxAgent's attempt counter

6.23.3.2 unsigned long int Langmuir::FluxAgent::attemptsSinceLast ( ) const

get the number of attempts since storeLast() was called

6.23.3.3 virtual double Langmuir::FluxAgent::energyChange (int *site*) [protected], [virtual]

The energy change associated with moving a carrier from the FluxAgent to a site.

Parameters

site	the site involved

Returns

the energy change

Reimplemented in Langmuir::ExcitonSourceAgent, Langmuir::HoleSourceAgent, Langmuir::ElectronSourceAgent, Langmuir::RecombinationAgent, Langmuir::HoleDrainAgent, and Langmuir::ElectronDrainAgent.

6.23.3.4 Grid::CubeFace Langmuir::FluxAgent::face ( ) const

get the Grid:CubeFace this FluxAgent is assigned to

6.23.3.5 QString Langmuir::FluxAgent::faceToLetter ( ) [protected]

convert the Grid::CubeFace to a single letter

For example, Grid::Left would return **L**. This is used in the output file titles.

6.23.3.6 Grid& Langmuir::FluxAgent::grid ( ) const

get the Grid this FluxAgent belongs to

6.23.3.7 void Langmuir::FluxAgent::initializeSite (int *site*) [protected]

assign the FluxAgent to a specific site in the grid

Parameters

site	the site in the grid

6.23.3.8 void Langmuir::FluxAgent::initializeSite (Grid::CubeFace *cubeFace*) [protected]

assign the FluxAgent to a specific Grid::CubeFace

Parameters

cubeFace	the face of a cubic grid; for example Grid::Left

When assigning to a specific Grid::CubeFace, the FluxAgent is considered to be a special agent, and thus resides in the sites reserved by the grid for special agents.

6.23.3.9 double Langmuir::FluxAgent::potential ( ) const

get the FluxAgent's potential

6.23.3.10 double Langmuir::FluxAgent::rate ( ) const

get the FluxAgent's rate

6.23.3.11 void Langmuir::FluxAgent::resetCounters ( )

set the attempt and success counters to zero

6.23.3.12 void Langmuir::FluxAgent::setAttempts (unsigned long int *value*)

set the FluxAgent's attempt counter

Parameters

potential	the value of the attempt counter

Warning

also calls storeLast()

6.23.3.13 void Langmuir::FluxAgent::setPotential (double *potential*)

set the FluxAgent's potential

Parameters

potential	the value of the potential

6.23.3.14 void Langmuir::FluxAgent::setRate (double *rate*)

set the FluxAgent's rate

Parameters

potential	the value of the rate

6.23.3.15 void Langmuir::FluxAgent::setRateSmartly (double *rate*, double *dflt*)

set the FluxAgent's rate

Parameters

potential	the value of the rate
dflt	the default value to set the rate to

If rate is negative, uses the default rate instead

6.23.3.16 void Langmuir::FluxAgent::setSuccesses (unsigned long int *value*)

set the FluxAgent's success counter

Parameters

potential	the value of the counter

Warning

also calls storeLast()

6.23.3.17 virtual bool Langmuir::FluxAgent::shouldTransport (int *site*) [protected], [virtual]

decide if the FluxAgent should transport a carrier to/from a given site

Parameters

site	the site involved

Returns

true if the FluxAgent should transport to/from the site

The default behavoir is for the FluxAgent to use a simple constant probabilty to make this decision. However, classes derived from FluxAgent can reimplement this function. For example, one might want to use a Metropolis criterion to make this decision.

Reimplemented in Langmuir::ExcitonSourceAgent, and Langmuir::SourceAgent.

6.23.3.18 unsigned long int Langmuir::FluxAgent::stepsSinceLast ( ) const

get the number of steps since storeLast() was called

6.23.3.19 void Langmuir::FluxAgent::storeLast ( )

set the value of last to the value of successes, and store the current step

6.23.3.20 unsigned long int Langmuir::FluxAgent::successes ( ) const

get the FluxAgent's success counter

6.23.3.21 unsigned long int Langmuir::FluxAgent::successesSinceLast ( ) const

get the number of successes since storeLast() was called

6.23.3.22 double Langmuir::FluxAgent::successProbability ( ) const

calculate and return the current probabilty of success

This is the number of successes divided by the number of attempts (x100). Ideally, this number should approach probability() as the simulation progresses, if shouldTransport() uses the simple constant probability method.

6.23.3.23 double Langmuir::FluxAgent::successProbabilitySinceLast ( ) const

calculate and return the probabilty of success since storeLast() was called

This is the number of successesSinceLast() divided by the number of attemptsSinceLast() (x100).

6.23.3.24 double Langmuir::FluxAgent::successRate ( ) const

calculate and return the current rate of success

This is the number of successes divided by the number of simulation steps. The current is related to the rate.

6.23.3.25 double Langmuir::FluxAgent::successRateSinceLast ( ) const

calculate and return the rate of success since storeLast() was called

This is the number of successesSinceLast() divided by the number of stepsSinceLast(). The current is related to the rate.

6.23.4 Member Data Documentation

6.23.4.1 unsigned long int Langmuir::FluxAgent::m_attempts [protected]

the number of times the FluxAgent has tried to transport.

6.23.4.2 Grid::CubeFace Langmuir::FluxAgent::m_face [protected]

the face of the grid this FluxAgent occupies

It may be Grid::NoFace is the FluxAgent occupies an actual site.

6.23.4.3 Grid& Langmuir::FluxAgent::m_grid [protected]

the grid this FluxAgent resides in

6.23.4.4 unsigned long int Langmuir::FluxAgent::m_lastAttempts [protected]

storage to note the number of successes at some step

6.23.4.5 unsigned long int Langmuir::FluxAgent::m_lastStep [protected]

the step at which last was noted

6.23.4.6 unsigned long int Langmuir::FluxAgent::m_lastSuccesses [protected]

storage to note the number of successes at some step

6.23.4.7 double Langmuir::FluxAgent::m_potential [protected]

the potential that is (possibly) used when calculating an energy change
The energy change can be used in the shouldTransport() function.

6.23.4.8 double Langmuir::FluxAgent::m_probability [protected]

the constant probability used in the default behavoir of shouldTransport().

6.23.4.9 unsigned long int Langmuir::FluxAgent::m_successes [protected]

the number of times the FluxAgent was successful in transporting.
The documentation for this class was generated from the following file:

- /home/adam/opt/langmuir/src/langmuirCore/include/fluxagent.h

6.24 Langmuir::FluxWriter Class Reference

A class to output source and drain info.
`#include <writer.h>`

Public Member Functions

- FluxWriter (World &world, const QString &name, QObject *parent=0)
  constructs the writer, has the same parameters as OutputInfo
- void write ()
  write the flux statistics of the current step to the stream

Protected Attributes

- World & m_world
  reference to the world object
- OutputStream m_stream
  output file stream

6.24.1 Detailed Description

A class to output source and drain info.

6.24.2 Constructor & Destructor Documentation

6.24.2.1 Langmuir::FluxWriter::FluxWriter (World & *world*, const QString & *name*, QObject * *parent* = 0)

constructs the writer, has the same parameters as OutputInfo

6.24.3 Member Function Documentation

6.24.3.1 void Langmuir::FluxWriter::write ( )

write the flux statistics of the current step to the stream

6.24.4 Member Data Documentation

6.24.4.1 OutputStream Langmuir::FluxWriter::m_stream [protected]

output file stream

6.24.4.2 World& Langmuir::FluxWriter::m_world [protected]

reference to the world object

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

- /home/adam/opt/langmuir/src/langmuirCore/include/writer.h

6.25 Grid Class Reference

A class to represent simulation grid.

`#include <grid.h>`

Public Slots

- virtual void makeConnections ()
  make signal/slot connections
- void setDimensions (int xsize, int ysize, int zsize)
  create the grid using dimensions
- void setColor (QColor color)
  set the color

Signals

- void colorChanged (QColor color)
  signal that the color of has changed
- void gridChanged ()
  signal that the grid has changed

Public Member Functions

- Grid (LangmuirViewer &viewer, QObject *parent=0)
  create the Grid
- ~Grid ()
  destroy the Grid
- const QColor & getColor () const
  get color

Protected Member Functions

- virtual void init ()
  initialize object
- virtual void draw ()
  perform OpenGL drawing operations
- void initShaders ()
  load the shaders
- void drawFallback ()
  render function
- void drawGrid ()
  render function

Protected Attributes

- QOpenGLShaderProgram m_shader1
  OpenGL shading pipeline.
- QOpenGLBuffer * m_verticesVBO
  vertices buffer
- unsigned int m_numPoints
  total number of points (# vertices / 3)
- QColor m_color
  color of grid
- bool m_shader1OK
  shader1 ok to use

6.25.1 Detailed Description

A class to represent simulation grid.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 Grid::Grid (LangmuirViewer & *viewer*, QObject * *parent* = 0) [explicit]

create the Grid

Parameters

viewer	the viewer
parent	QObject this belongs to

6.25.2.2 Grid::~Grid ( )

destroy the Grid

6.25.3 Member Function Documentation

6.25.3.1 void Grid::colorChanged (QColor *color*) [signal]

signal that the color of has changed

Parameters

color	value of color

6.25.3.2 virtual void Grid::draw ( ) [protected], [virtual]

perform OpenGL drawing operations

Reimplemented from SceneObject.

6.25.3.3 void Grid::drawFallback ( ) [protected]

render function

6.25.3.4 void Grid::drawGrid ( ) [protected]

render function

6.25.3.5 const QColor& Grid::getColor ( ) const

get color

6.25.3.6 void Grid::gridChanged ( ) [signal]

signal that the grid has changed

6.25.3.7 virtual void Grid::init ( ) [protected], [virtual]

initialize object

Reimplemented from SceneObject.

6.25.3.8 void Grid::initShaders ( ) [protected]

load the shaders

6.25.3.9 virtual void Grid::makeConnections ( ) [virtual], [slot]

make signal/slot connections

6.25.3.10 void Grid::setColor (QColor *color*) [slot]

set the color

Parameters

color	color to set

6.25.3.11 void Grid::setDimensions (int *xsize*, int *ysize*, int *zsize*) [slot]

create the grid using dimensions

Parameters

xsize	size of grid in x-direction
ysize	size of grid in y-direction
zsize	size of grid in z-direction

6.25.4 Member Data Documentation

6.25.4.1 QColor Grid::m_color [protected]

color of grid

6.25.4.2 unsigned int Grid::m_numPoints [protected]

total number of points (# vertices / 3)

6.25.4.3 QOpenGLShaderProgram Grid::m_shader1 [protected]

OpenGL shading pipeline.

6.25.4.4 bool Grid::m_shader1OK [protected]

shader1 ok to use

6.25.4.5 QOpenGLBuffer* Grid::m_verticesVBO [protected]

vertices buffer

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

- /home/adam/opt/langmuir/src/langmuirView/include/grid.h

6.26 Langmuir::Grid Class Reference

A class to hold Agents, calculate their positions, and store the background potential.

`#include <cubicgrid.h>`

Public Types

- enum CubeFace {
  Left = 0, Right = 1, Top = 2, Bottom = 3,
  Front = 4, Back = 5, NoFace = 6 }

  A way to indicate the faces of a cube.

Public Member Functions

- Grid (World &world, QObject *parent=0)

  Create a grid.
- ~Grid ()

  Destroy the grid.

- int xSize ()

  Get the number of sites along the x-direction.
- int ySize ()

  Get the number of sites along the y-direction.
- int zSize ()

  Get the number of sites along the z-direction.
- int xyPlaneArea ()

  Get the number of sites in the xy-plane.
- int volume ()

  Get the total number of sites.
- double totalDistance (int site1, int site2)

  Get the distance between two sites.
- double xDistance (int site1, int site2)

  Get the distance along the x-direction between two sites.
- double yDistance (int site1, int site2)

  Get the distance along the y-direction between two sites.
- double zDistance (int site1, int site2)

  Get the distance along the z-direction between two sites.
- double xImageDistance (int site1, int site2)

  Get the image distance along the x-direction between two sites.
- double yImageDistance (int site1, int site2)

  Get the image distance along the y-direction between two sites.
- double zImageDistance (int site1, int site2)

  Get the image distance along the z-direction between two sites.
- int xDistancei (int site1, int site2)

  *Get the **integer** distance along the x-direction between two sites.*
- int yDistancei (int site1, int site2)

  *Get the **integer** distance along the y-direction between two sites.*
- int zDistancei (int site1, int site2)

  *Get the **integer** distance along the z-direction between two sites.*
- int xImageDistancei (int site1, int site2)

  *Get the **integer** image distance along the x-direction between two sites.*
- int yImageDistancei (int site1, int site2)

  *Get the **integer** image distance along the y-direction between two sites.*
- int zImageDistancei (int site1, int site2)

  *Get the **integer** image distance along the z-direction between two sites.*
- int getIndexS (int xIndex, int yIndex, int zIndex=0)

  Get the serial site ID.
- int getIndexY (int site)

  Get the "y-site ID" from the "s-site ID".
- int getIndexX (int site)

  Get the "x-site ID" from the "s-site ID".
- int getIndexZ (int site)

  Get the "z-site ID" from the "s-site ID".
- double getPositionY (int site)

  Get the y-position from the "s-site ID".
- double getPositionX (int site)

  Get the x-position from the "s-site ID".
- double getPositionZ (int site)

  Get the z-position from the "s-site ID".
- Agent * agentAddress (int site)

Get a pointer to the Agent at a site.

- Agent::Type agentType (int site)

  Get the type of Agent at a site.
- void addToPotential (int site, double potential)

  Add some value to the background potential at a site.
- void setPotential (int site, double potential)

  Set the background potential at a site to some value.
- double potential (int site)

  Get the background potential at some site.
- QVector< int > neighborsSite (int site, int hoppingRange=1)

  Calculate the neighboring sites of a given site.
- QVector< int > neighborsFace (Grid::CubeFace cubeFace)

  Calculate the neighboring sites of a given face of the Grid.
- QVector< int > sliceIndex (int xi, int xf, int yi, int yf, int zi, int zf)

  Calculate the list of sites occupying a given range.
- void registerAgent (Agent *agent)

  Assign an Agent to a site in the Grid.
- void registerSpecialAgent (Agent *agent, Grid::CubeFace cubeFace)

  Assign an Agent to a special location.
- void unregisterAgent (Agent *agent)

  Remove an Agent from the Grid.
- void unregisterSpecialAgent (Agent *agent, Grid::CubeFace cubeFace)

  Remove an Agent from the special list of Agents in the Grid.
- void unregisterDefect (int site)

  Remove a defect from the Grid.
- void registerDefect (int site)

  Assign a site to be Agent::Defect.
- int specialAgentCount ()

  The total number of special Agents.
- QList< Agent * > & getSpecialAgentList (Grid::CubeFace cubeFace)

  Get a list of special Agents assigned to a specific Grid::CubeFace.

Static Public Member Functions

- static QString toQString (const Grid::CubeFace e)

Protected Attributes

- World & m_world

  Reference to the World object.
- QVector< Agent * > m_agents

  1D list of Agent pointers, the size of which is the volume of the Grid + the max number of special Agents.
- QVector< double > m_potentials

  1D list of site potentials, the size of which is the volume of the Grid + the max number of special Agents.
- QVector< Agent::Type > m_agentType

  1D list of Agent types, the size of which is the volume of the Grid + the max number of special Agents.
- QList< QList< Agent * > > m_specialAgents

  A list of lists of special agents, where each sub-list is for a different Grid::CubeFace.
- int m_specialAgentReserve

  The max number of special Agents allowed.

- int m_specialAgentCount

  The current number of special Agents registered with the Grid.
- int m_xSize

  The number of sites along the x-direction.
- int m_ySize

  The number of sites along the y-direction.
- int m_zSize

  The number of sites along the z-direction.
- int m_xyPlaneArea

  The number of sites in the xy-plane.
- int m_yzPlaneArea

  The number of sites in the yz-plane.
- int m_xzPlaneArea

  The number of sites in the xz-plane.
- int m_volume

  The total number of sites.

6.26.1 Detailed Description

A class to hold Agents, calculate their positions, and store the background potential.

The x-direction

- perpendicular to the electrodes
- runs from left to right
- corresponds to the dimension called **length**.

The y-direction

- parallel to the electrodes
- runs from bottom to top
- corresponds to the dimension called **width**.

The z-direction

- parallel to the electrodes
- runs from back to front
- corresponds to the dimension called **height**.

6.26.2 Member Enumeration Documentation

6.26.2.1 enum Langmuir::Grid::CubeFace

A way to indicate the faces of a cube.

Enumerator

Left x = 0, yz plane

Right x = lx, yz plane

Top z = 0, xy plane

Bottom z = lz, xy plane

Front y = 0, xz plane

Back y = ly, xz plane

NoFace undefined face

6.26.3 Constructor & Destructor Documentation

6.26.3.1 Langmuir::Grid::Grid (World & *world,* QObject * *parent* = 0)

Create a grid.

Parameters

world	reference to the world object
parent	QObject this belongs to

6.26.3.2 Langmuir::Grid::~Grid ( )

Destroy the grid.

6.26.4 Member Function Documentation

6.26.4.1 void Langmuir::Grid::addToPotential (int *site,* double *potential*)

Add some value to the background potential at a site.

Parameters

site	the "s-site ID"
potential	the value to add

6.26.4.2 Agent* Langmuir::Grid::agentAddress (int *site*)

Get a pointer to the Agent at a site.

Parameters

site	the "s-site ID"

Warning

may be NULL if there is no Agent

6.26.4.3 Agent::Type Langmuir::Grid::agentType (int *site*)

Get the type of Agent at a site.

Parameters

site	the "s-site ID"

Warning

if there is no Agent, it should be Agent::Empty

6.26.4.4 int Langmuir::Grid::getIndexS (int *xIndex*, int *yIndex*, int *zIndex* = 0)

Get the serial site ID.

Parameters

xIndex	x site ID
yIndex	y site ID
zIndex	z site ID

The position of a particle in the Grid can be thought of as a 3-tuple of (x, y, z) site IDs. However, this 3-tuple can be mapped/hashed into a single number using the dimension of the grid, called the "serial site ID", the "s-site ID", or just the "site".

6.26.4.5 int Langmuir::Grid::getIndexX (int *site*)

Get the "x-site ID" from the "s-site ID".

Parameters

site	the "s-site ID"

The y-site ID can be thought of as the x-value of the cornor of a Grid site.

See also

getIndexS

6.26.4.6 int Langmuir::Grid::getIndexY (int *site*)

Get the "y-site ID" from the "s-site ID".

Parameters

site	the "s-site ID"

The y-site ID can be thought of as the y-value of the cornor of a Grid site.

See also

getIndexS

6.26.4.7 int Langmuir::Grid::getIndexZ (int *site*)

Get the "z-site ID" from the "s-site ID".

Parameters

site	the "s-site ID"

The y-site ID can be thought of as the z-value of the cornor of a Grid site.

See also

getIndexS

6.26.4.8 double Langmuir::Grid::getPositionX (int *site*)

Get the x-position from the "s-site ID".

Parameters

site	the "s-site ID"

Particles are considered to reside in the "center" of Grid sites. The x-position is therefore the "x-site ID" plus 0.5 in reduced units.

6.26.4.9 double Langmuir::Grid::getPositionY (int *site*)

Get the y-position from the "s-site ID".

Parameters

site	the "s-site ID"

Particles are considered to reside in the "center" of Grid sites. The y-position is therefore the "y-site ID" plus 0.5 in reduced units.

6.26.4.10 double Langmuir::Grid::getPositionZ (int *site*)

Get the z-position from the "s-site ID".

Parameters

site	the "s-site ID"

Particles are considered to reside in the "center" of Grid sites. The z-position is therefore the "z-site ID" plus 0.5 in reduced units.

6.26.4.11 QList<Agent *>& Langmuir::Grid::getSpecialAgentList (Grid::CubeFace *cubeFace*)

Get a list of special Agents assigned to a specific Grid::CubeFace.

Parameters

cubeFace	the face of the Grid

6.26.4.12 QVector<int> Langmuir::Grid::neighborsFace (Grid::CubeFace *cubeFace*)

Calculate the neighboring sites of a given face of the Grid.

Parameters

cubeFace	the face of the Grid to consider

6.26.4.13 QVector<int> Langmuir::Grid::neighborsSite (int *site*, int *hoppingRange* = 1)

Calculate the neighboring sites of a given site.

Parameters

site	the "s-site ID"
hoppingRange	the number of adjacent sites to consider in the calculation

6.26.4.14 double Langmuir::Grid::potential (int *site*)

Get the background potential at some site.

Parameters

site	the "s-site ID"

6.26.4.15 void Langmuir::Grid::registerAgent (Agent * *agent*)

Assign an Agent to a site in the Grid.

Parameters

agent	a pointer to the Agent

Warning

uses Agent::getCurrentSite()
site must be Agent::Empty

Makes sure the site is empty first. After assigning the Agent to the site, calculates and assigns the neighbors to the Agent.

6.26.4.16 void Langmuir::Grid::registerDefect (int *site*)

Assign a site to be Agent::Defect.

Parameters

site	

6.26.4.17 void Langmuir::Grid::registerSpecialAgent (Agent * *agent,* Grid::CubeFace *cubeFace*)

Assign an Agent to a special location.

Parameters

agent	a pointer to the Agent
cubeFace	the face of the Grid

Agents such as Sources and Drains do not occupy a site in the Grid, and so must be stored in a special location.

6.26.4.18 void Langmuir::Grid::setPotential (int *site,* double *potential*)

Set the background potential at a site to some value.

Parameters

site	the "s-site ID"
potential	the value to set

6.26.4.19 QVector<int> Langmuir::Grid::sliceIndex (int *xi,* int *xf,* int *yi,* int *yf,* int *zi,* int *zf*)

Calculate the list of sites occupying a given range.

Parameters

xi	starting x-site ID
xf	stopping x-site ID
yi	starting y-site ID
yf	stopping y-site ID
zi	starting z-site ID
zf	stopping z-site ID

6.26.4.20 int Langmuir::Grid::specialAgentCount ( )

The total number of special Agents.

6.26.4.21 static QString Langmuir::Grid::toQString (const Grid::CubeFace *e*) [static]

6.26.4.22 double Langmuir::Grid::totalDistance (int *site1*, int *site2*)

Get the distance between two sites.

Parameters

site1	the first site
site2	the second site

6.26.4.23 void Langmuir::Grid::unregisterAgent (Agent * *agent*)

Remove an Agent from the Grid.

Parameters

agent	a pointer to the Agent

6.26.4.24 void Langmuir::Grid::unregisterDefect (int *site*)

Remove a defect from the Grid.

Parameters

site	the "s-site ID"

6.26.4.25 void Langmuir::Grid::unregisterSpecialAgent (Agent * *agent*, Grid::CubeFace *cubeFace*)

Remove an Agent from the special list of Agents in the Grid.

Parameters

agent	a pointer to the Agent
cubeFace	the face of the Grid

6.26.4.26 int Langmuir::Grid::volume ( )

Get the total number of sites.

6.26.4.27 double Langmuir::Grid::xDistance (int *site1,* int *site2*)

Get the distance along the x-direction between two sites.

Parameters

site1	the first site
site2	the second site

6.26.4.28 int Langmuir::Grid::xDistancei (int *site1*, int *site2*)

Get the **integer** distance along the x-direction between two sites.

Parameters

site1	the first site
site2	the second site

6.26.4.29 double Langmuir::Grid::xImageDistance (int *site1*, int *site2*)

Get the image distance along the x-direction between two sites.

Parameters

site1	the first site
site2	the second site (reflected)

The second site's x-position is taken to be the negative of its x-value (i.e., the particle is reflected through the yz-plane).

6.26.4.30 int Langmuir::Grid::xImageDistancei (int *site1*, int *site2*)

Get the **integer** image distance along the x-direction between two sites.

Parameters

site1	the first site
site2	the second site (reflected)

The second site's x-position is taken to be the negative of its x-value (i.e., the particle is reflected through the yz-plane).

6.26.4.31 int Langmuir::Grid::xSize ( )

Get the number of sites along the x-direction.

6.26.4.32 int Langmuir::Grid::xyPlaneArea ( )

Get the number of sites in the xy-plane.

6.26.4.33 double Langmuir::Grid::yDistance (int *site1*, int *site2*)

Get the distance along the y-direction between two sites.

Parameters

site1	the first site

site2	the second site

6.26.4.34 int Langmuir::Grid::yDistancei (int *site1*, int *site2*)

Get the **integer** distance along the y-direction between two sites.

Parameters

site1	the first site
site2	the second site

6.26.4.35 double Langmuir::Grid::yImageDistance (int *site1*, int *site2*)

Get the image distance along the y-direction between two sites.

Parameters

site1	the first site
site2	the second site (reflected)

The second site's y-position is taken to be the negative of its y-value (i.e., the particle is reflected through the xz-plane).

6.26.4.36 int Langmuir::Grid::yImageDistancei (int *site1*, int *site2*)

Get the **integer** image distance along the y-direction between two sites.

Parameters

site1	the first site
site2	the second site (reflected)

The second site's y-position is taken to be the negative of its y-value (i.e., the particle is reflected through the xz-plane).

6.26.4.37 int Langmuir::Grid::ySize ( )

Get the number of sites along the y-direction.

6.26.4.38 double Langmuir::Grid::zDistance (int *site1*, int *site2*)

Get the distance along the z-direction between two sites.

Parameters

site1	the first site
site2	the second site

6.26.4.39 int Langmuir::Grid::zDistancei (int *site1*, int *site2*)

Get the **integer** distance along the z-direction between two sites.

Parameters

site1	the first site
site2	the second site

6.26.4.40 double Langmuir::Grid::zImageDistance (int *site1*, int *site2*)

Get the image distance along the z-direction between two sites.

Parameters

site1	the first site
site2	the second site (reflected)

The second site's z-position is taken to be the negative of its z-value (i.e., the particle is reflected through the xy-plane).

6.26.4.41 int Langmuir::Grid::zImageDistancei (int *site1*, int *site2*)

Get the **integer** image distance along the z-direction between two sites.

Parameters

site1	the first site
site2	the second site (reflected)

The second site's z-position is taken to be the negative of its z-value (i.e., the particle is reflected through the xy-plane).

6.26.4.42 int Langmuir::Grid::zSize ( )

Get the number of sites along the z-direction.

6.26.5 Member Data Documentation

6.26.5.1 QVector<Agent *> Langmuir::Grid::m_agents [protected]

1D list of Agent pointers, the size of which is the volume of the Grid + the max number of special Agents.

Warning

some of these may be NULL

Each position in the list is mapped to a position in the Grid. Use getIndexS() to calculate the serial site ID needed to index this list.

6.26.5.2 QVector<Agent::Type> Langmuir::Grid::m_agentType [protected]

1D list of Agent types, the size of which is the volume of the Grid + the max number of special Agents.

Warning

some of these may be Agent::Empty

Each position in the list is mapped to a position in the Grid. Use getIndexS() to calculate the serial site ID needed to index this list.

6.26.5.3 QVector<double> Langmuir::Grid::m_potentials [protected]

1D list of site potentials, the size of which is the volume of the Grid + the max number of special Agents.

Each position in the list is mapped to a position in the Grid. Use getIndexS() to calculate the serial site ID needed to index this list.

6.26.5.4 int Langmuir::Grid::m_specialAgentCount [protected]

The current number of special Agents registered with the Grid.

6.26.5.5 int Langmuir::Grid::m_specialAgentReserve [protected]

The max number of special Agents allowed.

6.26.5.6 QList< QList<Agent *> > Langmuir::Grid::m_specialAgents [protected]

A list of lists of special agents, where each sub-list is for a different Grid::CubeFace.

6.26.5.7 int Langmuir::Grid::m_volume [protected]

The total number of sites.

6.26.5.8 World& Langmuir::Grid::m_world [protected]

Reference to the World object.

6.26.5.9 int Langmuir::Grid::m_xSize [protected]

The number of sites along the x-direction.

6.26.5.10 int Langmuir::Grid::m_xyPlaneArea [protected]

The number of sites in the xy-plane.

6.26.5.11 int Langmuir::Grid::m_xzPlaneArea [protected]

The number of sites in the xz-plane.

6.26.5.12 int Langmuir::Grid::m_ySize [protected]

The number of sites along the y-direction.

6.26.5.13 int Langmuir::Grid::m_yzPlaneArea [protected]

The number of sites in the yz-plane.

6.26.5.14 int Langmuir::Grid::m_zSize [protected]

The number of sites along the z-direction.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/cubicgrid.h

6.27 Langmuir::GridImage Class Reference

A class to draw images of the grid.

`#include <writer.h>`

Public Member Functions

- GridImage (World &world, QColor bg=Qt::black, QObject *parent=0)
  create the image and painter, setting the background and size
- void drawSites (QList< int > &sites, QColor color, int layer)
  draw some sites
- void drawCharges (QList< ChargeAgent * > &charges, QColor color, int layer)
  draw some sites
- void save (QString name, int scale=3)
  save the image to a file

Private Attributes

- QPainter m_painter
  the painter that paints the image
- QImage m_image
  the image we draw onto
- World & m_world
  reference to the world object

6.27.1 Detailed Description

A class to draw images of the grid.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 Langmuir::GridImage::GridImage (World & *world*, QColor *bg* = `Qt::black`, QObject * *parent* = `0`)

create the image and painter, setting the background and size

Parameters

world	Reference to the world object
bg	Background color

parent	parent QObject

6.27.3 Member Function Documentation

6.27.3.1 void Langmuir::GridImage::drawCharges (QList< ChargeAgent * > & *charges*, QColor *color*, int *layer*)

draw some sites

Parameters

charges	A list of ChargeAgents, which have site ids • could be the list of electrons • could be the list of holes
color	The color of the points
layer	Which layer are we drawing? its a 2D image

6.27.3.2 void Langmuir::GridImage::drawSites (QList< int > & *sites*, QColor *color*, int *layer*)

draw some sites

Parameters

sites	A list of integers that are site ids • could be the list of trap ids • could be the list of defect ids
color	The color of the points
layer	Which layer are we drawing? its a 2D image

6.27.3.3 void Langmuir::GridImage::save (QString *name*, int *scale* = 3)

save the image to a file

Parameters

name	A file name that is passed to a OutputInfo object, the output is assummed png
scale	Multiply the image by some scale, increasing the resolution

6.27.4 Member Data Documentation

6.27.4.1 QImage Langmuir::GridImage::m_image [private]

the image we draw onto

6.27.4.2 QPainter Langmuir::GridImage::m_painter [private]

the painter that paints the image

6.27.4.3 World& Langmuir::GridImage::m_world [private]

reference to the world object

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

- /home/adam/opt/langmuir/src/langmuirCore/include/writer.h

6.28 Langmuir::GridViewGL Class Reference

`#include <gridview.h>`

Public Slots

- void setXTranslation (double length)
- void setYTranslation (double length)
- void setZTranslation (double length)
- void setXRotation (double angle)
- void setYRotation (double angle)
- void setZRotation (double angle)
- void setTimerInterval (int time)
- void setIterationsPrint (int iterationsPrint)
- void toggleCoulombStatus (int checkState)
- void toggleOpenCLStatus (int checkState)
- void toggleTrapsTexture (int checkState)
- void togglePauseStatus ()
- void toggleRecording ()
- void timerUpdateGL ()
- void timerRecordShot ()
- void screenShot ()
- void resetView ()
- void updatePointBuffers ()

Signals

- void xTranslationChanged (double angle)
- void yTranslationChanged (double angle)
- void zTranslationChanged (double angle)
- void xRotationChanged (double angle)
- void yRotationChanged (double angle)
- void zRotationChanged (double angle)
- void statusMessage (QString, int time=0)
- void pauseChanged (QString)
- void recordChanged (QString)
- void recordChangedColor (QColor)
- void openCLStatusChanged (int checkState)
- void coulombStatusChanged (int checkState)
- void iterationsPrintChanged (int iterationsPrint)
- void timerIntervalChanged (int time)
- void stepChanged (int step)
- void currentChanged (double current)
- void initialized ()

Public Member Functions

- GridViewGL (const QGLFormat &format, QWidget *parent, QString input)
- ~GridViewGL ()
- QSize minimumSizeHint () const
- QSize sizeHint () const
- QImage drawEnergyLandscape (int layer=0)
- void NormalizeAngle (double &angle)

Public Attributes

- QVector3D translation
- QVector3D rotation
- QVector3D delta
- float thickness
- float fov
- QPoint lastPos
- QTimer * qtimer
- Box * base
- Box * source
- Box * drain
- Box * side1
- Box * side2
- Box * side3
- Box * side4
- Box * side5
- Box * side6
- ColoredObject * background
- ColoredObject * ambientLight
- ColoredObject * diffuseLight
- ColoredObject * specularLight
- QVector< float > pointBuffer1
- QVector< float > pointBuffer2
- PointArray * carriersMinus
- PointArray * carriersPlus
- PointArray * defects
- Simulation * pSim
- World * pWorld
- bool pause
- bool recording
- QTimer * recordTimer
- RecordDialog * recordDialog
- int updateTime
- int step
- GLuint trapsTexture
- GLuint metalTexture

Protected Member Functions

- void initializeGL ()
- void resizeGL (int w, int h)
- void paintGL ()
- void mousePressEvent (QMouseEvent *event)
- void mouseMoveEvent (QMouseEvent *event)
- void wheelEvent (QWheelEvent *event)
- void keyPressEvent (QKeyEvent *event)

6.28.1 Constructor & Destructor Documentation

6.28.1.1 Langmuir::GridViewGL::GridViewGL (const QGLFormat & *format,* QWidget * *parent,* QString *input*)

6.28.1.2 Langmuir::GridViewGL::~GridViewGL ( )

6.28.2 Member Function Documentation

6.28.2.1 void Langmuir::GridViewGL::coulombStatusChanged (int *checkState*) [signal]

6.28.2.2 void Langmuir::GridViewGL::currentChanged (double *current*) [signal]

6.28.2.3 QImage Langmuir::GridViewGL::drawEnergyLandscape (int *layer* = 0)

6.28.2.4 void Langmuir::GridViewGL::initialized ( ) [signal]

6.28.2.5 void Langmuir::GridViewGL::initializeGL ( ) [protected]

6.28.2.6 void Langmuir::GridViewGL::iterationsPrintChanged (int *iterationsPrint*) [signal]

6.28.2.7 void Langmuir::GridViewGL::keyPressEvent (QKeyEvent * *event*) [protected]

6.28.2.8 QSize Langmuir::GridViewGL::minimumSizeHint ( ) const

6.28.2.9 void Langmuir::GridViewGL::mouseMoveEvent (QMouseEvent * *event*) [protected]

6.28.2.10 void Langmuir::GridViewGL::mousePressEvent (QMouseEvent * *event*) [protected]

6.28.2.11 void Langmuir::GridViewGL::NormalizeAngle (double & *angle*)

6.28.2.12 void Langmuir::GridViewGL::openCLStatusChanged (int *checkState*) [signal]

6.28.2.13 void Langmuir::GridViewGL::paintGL ( ) [protected]

6.28.2.14 void Langmuir::GridViewGL::pauseChanged (QString) [signal]

6.28.2.15 void Langmuir::GridViewGL::recordChanged (QString) [signal]

6.28.2.16 void Langmuir::GridViewGL::recordChangedColor (QColor) [signal]

6.28.2.17 void Langmuir::GridViewGL::resetView ( ) [slot]

6.28.2.18 void Langmuir::GridViewGL::resizeGL (int *w,* int *h*) [protected]

6.28.2.19 void Langmuir::GridViewGL::screenShot ( ) [slot]

6.28.2.20 void Langmuir::GridViewGL::setIterationsPrint (int *iterationsPrint*) [slot]

6.28.2.21 void Langmuir::GridViewGL::setTimerInterval (int *time*) [slot]

6.28.2.22 void Langmuir::GridViewGL::setXRotation (double *angle*) [slot]

6.28.2.23 void Langmuir::GridViewGL::setXTranslation (double *length*) [slot]

6.28.2.24 void Langmuir::GridViewGL::setYRotation (double *angle*) [slot]

6.28.2.25 void Langmuir::GridViewGL::setYTranslation (double *length*) [slot]

6.28.2.26 void Langmuir::GridViewGL::setZRotation (double *angle*) [slot]

6.28.2.27 void Langmuir::GridViewGL::setZTranslation (double *length*) [slot]

6.28.2.28 QSize Langmuir::GridViewGL::sizeHint ( ) const

6.28.2.29 void Langmuir::GridViewGL::statusMessage (QString , int *time* = 0) [signal]

6.28.2.30 void Langmuir::GridViewGL::stepChanged (int *step*) [signal]

6.28.2.31 void Langmuir::GridViewGL::timerIntervalChanged (int *time*) [signal]

6.28.2.32 void Langmuir::GridViewGL::timerRecordShot ( ) [slot]

6.28.2.33 void Langmuir::GridViewGL::timerUpdateGL ( ) [slot]

6.28.2.34 void Langmuir::GridViewGL::toggleCoulombStatus (int *checkState*) [slot]

6.28.2.35 void Langmuir::GridViewGL::toggleOpenCLStatus (int *checkState*) [slot]

6.28.2.36 void Langmuir::GridViewGL::togglePauseStatus ( ) [slot]

6.28.2.37 void Langmuir::GridViewGL::toggleRecording ( ) [slot]

6.28.2.38 void Langmuir::GridViewGL::toggleTrapsTexture (int *checkState*) [slot]

6.28.2.39 void Langmuir::GridViewGL::updatePointBuffers ( ) [slot]

6.28.2.40 void Langmuir::GridViewGL::wheelEvent (QWheelEvent * *event*) [protected]

6.28.2.41 void Langmuir::GridViewGL::xRotationChanged (double *angle*) [signal]

6.28.2.42 void Langmuir::GridViewGL::xTranslationChanged (double *angle*) [signal]

6.28.2.43 void Langmuir::GridViewGL::yRotationChanged (double *angle*) [signal]

6.28.2.44 void Langmuir::GridViewGL::yTranslationChanged (double *angle*) [signal]

6.28.2.45 void Langmuir::GridViewGL::zRotationChanged (double *angle*) [signal]

6.28.2.46 void Langmuir::GridViewGL::zTranslationChanged (double *angle*) [signal]

6.28.3 Member Data Documentation

6.28.3.1 **ColoredObject*** Langmuir::GridViewGL::ambientLight

6.28.3.2 **ColoredObject*** Langmuir::GridViewGL::background

6.28.3.3 **Box*** Langmuir::GridViewGL::base

6.28.3.4 **PointArray*** Langmuir::GridViewGL::carriersMinus

6.28.3.5 **PointArray*** Langmuir::GridViewGL::carriersPlus

6.28.3.6 **PointArray*** Langmuir::GridViewGL::defects

6.28.3.7 QVector3D Langmuir::GridViewGL::delta

6.28.3.8 **ColoredObject*** Langmuir::GridViewGL::diffuseLight

6.28.3.9 **Box*** Langmuir::GridViewGL::drain

6.28.3.10 float Langmuir::GridViewGL::fov

6.28.3.11 QPoint Langmuir::GridViewGL::lastPos

6.28.3.12 GLuint Langmuir::GridViewGL::metalTexture

6.28.3.13 bool Langmuir::GridViewGL::pause

6.28.3.14 QVector<float> Langmuir::GridViewGL::pointBuffer1

6.28.3.15 QVector<float> Langmuir::GridViewGL::pointBuffer2

6.28.3.16 **Simulation*** Langmuir::GridViewGL::pSim

6.28.3.17 **World*** Langmuir::GridViewGL::pWorld

6.28.3.18 QTimer* Langmuir::GridViewGL::qtimer

6.28.3.19 **RecordDialog*** Langmuir::GridViewGL::recordDialog

6.28.3.20 bool Langmuir::GridViewGL::recording

6.28.3.21 QTimer* Langmuir::GridViewGL::recordTimer

6.28.3.22 QVector3D Langmuir::GridViewGL::rotation

6.28.3.23 **Box*** Langmuir::GridViewGL::side1

6.28.3.24 **Box*** Langmuir::GridViewGL::side2

6.28.3.25 **Box*** Langmuir::GridViewGL::side3

6.28.3.26 **Box*** Langmuir::GridViewGL::side4

6.28.3.27 **Box*** Langmuir::GridViewGL::side5

6.28.3.28 **Box*** Langmuir::GridViewGL::side6

6.28.3.29 **Box*** Langmuir::GridViewGL::source

6.28.3.30 **ColoredObject*** Langmuir::GridViewGL::specularLight

6.28.3.31 int Langmuir::GridViewGL::step

6.28.3.32 float Langmuir::GridViewGL::thickness

6.28.3.33 QVector3D Langmuir::GridViewGL::translation

6.28.3.34 GLuint Langmuir::GridViewGL::trapsTexture

6.28.3.35 int Langmuir::GridViewGL::updateTime

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.29 Langmuir::HoleAgent Class Reference

A class to represent moving positive charges.

`#include <chargeagent.h>`

Public Member Functions

- HoleAgent (World &world, int site, QObject *parent=0)
  Construct HoleAgent.

Protected Member Functions

- virtual double bindingPotential (int site)
  Calculate Exciton Binding Energy.
- virtual Agent::Type otherType ()
  Return other Agent::Type.
- virtual Grid & otherGrid ()
  Return other Grid.

Additional Inherited Members

6.29.1 Detailed Description

A class to represent moving positive charges.

6.29.2 Constructor & Destructor Documentation

6.29.2.1 Langmuir::HoleAgent::HoleAgent (World & *world,* int *site,* QObject * *parent* = `0`)

Construct HoleAgent.

6.29.3 Member Function Documentation

6.29.3.1 virtual double Langmuir::HoleAgent::bindingPotential (int *site*) [protected], [virtual]

Calculate Exciton Binding Energy.

Parameters

site	the site to check in other Grid

Returns

- -0.5 eV if exciton
- 0 otherwise

Implements Langmuir::ChargeAgent.

6.29.3.2 virtual Grid& Langmuir::HoleAgent::otherGrid ( ) [protected], [virtual]

Return other Grid.

Returns

World::electronGrid

Implements Langmuir::ChargeAgent.

6.29.3.3 virtual Agent::Type Langmuir::HoleAgent::otherType ( ) [protected], [virtual]

Return other Agent::Type.

Returns

Agent::Electron

Implements Langmuir::ChargeAgent.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/chargeagent.h

6.30 Langmuir::HoleDrainAgent Class Reference

A class to remove HoleAgents.

`#include <drainagent.h>`

Public Member Functions

- HoleDrainAgent (World &world, int site, QObject *parent=0)
  create an HoleDrainAgent at a specific site
- HoleDrainAgent (World &world, Grid::CubeFace cubeFace, QObject *parent=0)
  create a HoleDrainAgent at a specific Grid::CubeFace

Protected Member Functions

- virtual double energyChange (int fSite)
  same as FluxAgent::energyChange(), but specialized for HoleAgents.

Additional Inherited Members

6.30.1 Detailed Description

A class to remove HoleAgents.

6.30.2 Constructor & Destructor Documentation

6.30.2.1 Langmuir::HoleDrainAgent::HoleDrainAgent (World & *world,* int *site,* QObject * *parent* = 0)

create an HoleDrainAgent at a specific site

6.30.2.2 Langmuir::HoleDrainAgent::HoleDrainAgent (World & *world,* Grid::CubeFace *cubeFace,* QObject * *parent* = 0)

create a HoleDrainAgent at a specific Grid::CubeFace

6.30.3 Member Function Documentation

6.30.3.1 virtual double Langmuir::HoleDrainAgent::energyChange (int *fSite*) [protected], [virtual]

same as FluxAgent::energyChange(), but specialized for HoleAgents.

Note really used because the default FluxAgent::shouldTransport() behavoir, which is to use a simple constant probability, has not been reimplemented for DrainAgents.

Reimplemented from Langmuir::FluxAgent.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/drainagent.h

6.31 Langmuir::HoleSourceAgent Class Reference

A class to inject HoleAgents.

`#include <sourceagent.h>`

Public Member Functions

- HoleSourceAgent (World &world, int site, QObject *parent=0)
  create a HoleSourceAgent at a specific site
- HoleSourceAgent (World &world, Grid::CubeFace cubeFace, QObject *parent=0)
  create a HoleSourceAgent at a specific Grid::CubeFace

Protected Member Functions

- virtual bool validToInject (int site)
  same as SourceAgent::validToInject(), but specialized for HoleAgents.
- virtual double energyChange (int site)
  same as FluxAgent::energyChange(), but specialized for HoleAgents.
- virtual void inject (int site)
  same as SourceAgent::inject(), but specialized for HoleAgents.

Additional Inherited Members

6.31.1 Detailed Description

A class to inject HoleAgents.

6.31.2 Constructor & Destructor Documentation

6.31.2.1 Langmuir::HoleSourceAgent::HoleSourceAgent (World & *world,* int *site,* QObject * *parent* = 0)

create a HoleSourceAgent at a specific site

6.31.2.2 Langmuir::HoleSourceAgent::HoleSourceAgent (World & *world,* Grid::CubeFace *cubeFace,* QObject * *parent* = 0)

create a HoleSourceAgent at a specific Grid::CubeFace

6.31.3 Member Function Documentation

6.31.3.1 virtual double Langmuir::HoleSourceAgent::energyChange (int *site*) [protected], [virtual]

same as FluxAgent::energyChange(), but specialized for HoleAgents.

Reimplemented from Langmuir::FluxAgent.

6.31.3.2 virtual void Langmuir::HoleSourceAgent::inject (int *site*) [protected], [virtual]

same as SourceAgent::inject(), but specialized for HoleAgents.

Implements Langmuir::SourceAgent.

6.31.3.3 virtual bool Langmuir::HoleSourceAgent::validToInject (int *site*) [protected], [virtual]

same as SourceAgent::validToInject(), but specialized for HoleAgents.

Implements Langmuir::SourceAgent.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/sourceagent.h

6.32 MarchingCubes::Isosurface Class Reference

A class to compute a contour iso-surface.

`#include <isosurface.h>`

Signals

- void done ()

  signal that the calculation is done
- void progress (int)

  signal progress

Public Member Functions

- Isosurface (QObject *parent=0)
  Create isosurface.
- ~Isosurface ()
  Destroy isosurface.
- scalar_field & createScalarField (int xsize, int ysize, int zsize, float spacing=1.0)
  set up scalar field
- scalar_field * getScalarField ()
  get reference to scalar field
- void setIsoValue (float value)
  set contour isovalue
- void generate ()
  perform calculation
- void clear ()
  clear data
- const QVector< float > & vertices () const
  get list of vertices
- const QVector< float > & normals () const
  get list of normals
- const QVector< unsigned int > & indices () const
  get list of indices

Private Member Functions

- float getOffset (const float &v1, const float &v2, const float &v)
  find the approximate point of intersection of the surface between two points with the values v1 and v2
- void marchingCubes (int xi, int yi, int zi)
  perform marching cubes algorithm at array point
- void simplify ()
  simplify vertex, normal, and index arrays

Private Attributes

- QList< Triangle * > m_triangles
  list of triangles
- QVector< float > m_vertices
  list of vertices
- QVector< float > m_normals
  list of normals
- QVector< unsigned int > m_indices
  list of indices
- scalar_field * m_scalar
  3D array of field
- float m_spacing
  grid spacing
- float m_value
  contour iso-value
- int m_xsize
  grid points (x)

- int m_ysize
  grid points (y)
- int m_zsize
  grid points (z)

6.32.1 Detailed Description

A class to compute a contour iso-surface.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 MarchingCubes::Isosurface::Isosurface (QObject * *parent* = 0) [explicit]

Create isosurface.

6.32.2.2 MarchingCubes::Isosurface::~Isosurface ( )

Destroy isosurface.

6.32.3 Member Function Documentation

6.32.3.1 void MarchingCubes::Isosurface::clear ( )

clear data

6.32.3.2 scalar_field& MarchingCubes::Isosurface::createScalarField (int *xsize*, int *ysize*, int *zsize*, float *spacing* = 1.0)

set up scalar field

Parameters

xsize	number of x points
ysize	number of y points
zsize	number of z points
spacing	grid spacing

6.32.3.3 void MarchingCubes::Isosurface::done ( ) [signal]

signal that the calculation is done

6.32.3.4 void MarchingCubes::Isosurface::generate ( )

perform calculation

6.32.3.5 float MarchingCubes::Isosurface::getOffset (const float & *v1*, const float & *v2*, const float & *v*) [private]

find the approximate point of intersection of the surface between two points with the values v1 and v2

6.32.3.6 scalar_field* MarchingCubes::Isosurface::getScalarField ( )

get reference to scalar field

6.32.3.7 const QVector<unsigned int>& MarchingCubes::Isosurface::indices ( ) const

get list of indices

6.32.3.8 void MarchingCubes::Isosurface::marchingCubes (int *xi,* int *yi,* int *zi*) [private]

perform marching cubes algorithm at array point

6.32.3.9 const QVector<float>& MarchingCubes::Isosurface::normals ( ) const

get list of normals

6.32.3.10 void MarchingCubes::Isosurface::progress (int) [signal]

signal progress

6.32.3.11 void MarchingCubes::Isosurface::setIsoValue (float *value*)

set contour isovalue

Parameters

value	value to set

6.32.3.12 void MarchingCubes::Isosurface::simplify ( ) [private]

simplify vertex, normal, and index arrays

6.32.3.13 const QVector<float>& MarchingCubes::Isosurface::vertices ( ) const

get list of vertices

6.32.4 Member Data Documentation

6.32.4.1 QVector<unsigned int> MarchingCubes::Isosurface::m_indices [private]

list of indices

6.32.4.2 QVector<float> MarchingCubes::Isosurface::m_normals [private]

list of normals

6.32.4.3 scalar_field* MarchingCubes::Isosurface::m_scalar [private]

3D array of field

6.32.4.4 float MarchingCubes::Isosurface::m_spacing [private]

grid spacing

6.32.4.5 QList<Triangle*> MarchingCubes::Isosurface::m_triangles [private]

list of triangles

6.32.4.6 float MarchingCubes::Isosurface::m_value [private]

contour iso-value

6.32.4.7 QVector<float> MarchingCubes::Isosurface::m_vertices [private]

list of vertices

6.32.4.8 int MarchingCubes::Isosurface::m_xsize [private]

grid points (x)

6.32.4.9 int MarchingCubes::Isosurface::m_ysize [private]

grid points (y)

6.32.4.10 int MarchingCubes::Isosurface::m_zsize [private]

grid points (z)

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

- /home/adam/opt/langmuir/src/langmuirView/include/isosurface.h

6.33 IsoSurfaceDialog Class Reference

`#include <isosurfacedialog.h>`

Public Slots

- void init ()
- void update ()
- void on_calculateButton_clicked ()
- void setCalculateEnabled (bool enabled)
- void setProgressRange (int low, int high)
- void setProgress (int value)
- void updateComboBoxMode (Mesh::Mode mode)
- void updateSpinBoxAlpha (int value)
- void on_comboBoxMode_currentTextChanged (const QString &text)
- void on_spinBoxAlpha_valueChanged (int value)
- void extractAlpha (QColor color)

Signals

- void sendAlpha (int value)

Public Member Functions

- IsoSurfaceDialog (LangmuirViewer &viewer, QWidget *parent=0)
- ~IsoSurfaceDialog ()

Private Attributes

- Ui::IsoSurfaceDialog * ui
- LangmuirViewer & m_viewer

6.33.1 Constructor & Destructor Documentation

6.33.1.1 IsoSurfaceDialog::IsoSurfaceDialog (LangmuirViewer & *viewer,* QWidget * *parent* = 0) [explicit]

6.33.1.2 IsoSurfaceDialog::~IsoSurfaceDialog ( )

6.33.2 Member Function Documentation

6.33.2.1 void IsoSurfaceDialog::extractAlpha (QColor *color*) [slot]

6.33.2.2 void IsoSurfaceDialog::init ( ) [slot]

6.33.2.3 void IsoSurfaceDialog::on_calculateButton_clicked ( ) [slot]

6.33.2.4 void IsoSurfaceDialog::on_comboBoxMode_currentTextChanged (const QString & *text*) [slot]

6.33.2.5 void IsoSurfaceDialog::on_spinBoxAlpha_valueChanged (int *value*) [slot]

6.33.2.6 void IsoSurfaceDialog::sendAlpha (int *value*) [signal]

6.33.2.7 void IsoSurfaceDialog::setCalculateEnabled (bool *enabled*) [slot]

6.33.2.8 void IsoSurfaceDialog::setProgress (int *value*) [slot]

6.33.2.9 void IsoSurfaceDialog::setProgressRange (int *low,* int *high*) [slot]

6.33.2.10 void IsoSurfaceDialog::update ( ) [slot]

6.33.2.11 void IsoSurfaceDialog::updateComboBoxMode (Mesh::Mode *mode*) [slot]

6.33.2.12 void IsoSurfaceDialog::updateSpinBoxAlpha (int *value*) [slot]

6.33.3 Member Data Documentation

6.33.3.1 LangmuirViewer& IsoSurfaceDialog::m_viewer [private]

6.33.3.2 Ui::IsoSurfaceDialog* IsoSurfaceDialog::ui [private]

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

- /home/adam/opt/langmuir/src/langmuirView/include/isosurfacedialog.h

6.34 Langmuir::KeyValueParser Class Reference

A class to read the parameters and store them in the correct place.

`#include <keyvalueparser.h>`

Public Member Functions

- KeyValueParser (World &world, QObject *parent=0)
  Create a KeyValueParser.
- ~KeyValueParser ()
  Destroy the KeyValueParser.
- SimulationParameters & parameters ()
  Get the SimulationParameters.
- void parse (const QString &line)
  Parse a string and assign a value to the correct parameter.
- void save (const QString &fileName="%stub.parm")
  Write the parameters to a file in a "key=value" fashion.
- Variable & getVariable (const QString &key)
  Get a reference to a variable by name.
- const QMap< QString, Variable * > & getVariableMap () const
  get the variable map
- const QStringList & getOrderedNames () const
  get list of ordered keys
- QString toQString ()
  convert parameters to a QString

Private Member Functions

- template<typename T >
  void registerVariable (const QString &key, T &value, Variable::VariableMode mode=0)
  Register an allowed variable with the parser.

Private Attributes

- QMap< QString, Variable * > m_variableMap
  A map between variable names and variables.
- QStringList m_orderedNames
  A list of variable names in a specific order.
- SimulationParameters m_parameters
  The simulation parameters.
- World & m_world
  Reference to World object.

Friends

- std::ostream & operator<< (std::ostream &stream, const KeyValueParser &keyValueParser)
  Write the parameters to a std::ostream in a "key=value" fashion.
- QDebug operator<< (QDebug dbg, KeyValueParser &keyValueParser)
  Write the parameters to QDebug in a "key=value" fashion.

6.34.1 Detailed Description

A class to read the parameters and store them in the correct place.

The location of the SimulationParameters object for the entire simulation is a private variable of this class.

To add new variables, follow these steps:

- declare the new variable in the SimulationParameters struct (parameters.h)
- assign the default value of the new variable in the SimulationParameters constructor (parameters.h)
- implement validity checking for the variable in the checkSimulationParameters() function (parameters.h)
- register the variable in the KeyValueParser constructor using the registerVariable() function (keyvalueparser.h)
- to use non-standard types, you must overload certain template functions in variable.h. See, for example, overloads for QDateTime in variable.h.

6.34.2 Constructor & Destructor Documentation

6.34.2.1 Langmuir::KeyValueParser::KeyValueParser (World & *world*, QObject * *parent* = 0)

Create a KeyValueParser.

Parameters

world	reference to World Object
parent	QObject this belongs to

Add calls to registerVariable() to add new variables to the simulation.

6.34.2.2 Langmuir::KeyValueParser::~KeyValueParser ( )

Destroy the KeyValueParser.

6.34.3 Member Function Documentation

6.34.3.1 const QStringList& Langmuir::KeyValueParser::getOrderedNames ( ) const

get list of ordered keys

6.34.3.2 Variable& Langmuir::KeyValueParser::getVariable (const QString & *key*)

Get a reference to a variable by name.

Parameters

name	the name of the variable

6.34.3.3 const QMap<QString,Variable*>& Langmuir::KeyValueParser::getVariableMap ( ) const

get the variable map

6.34.3.4 SimulationParameters& Langmuir::KeyValueParser::parameters ( )

Get the SimulationParameters.

6.34.3.5 void Langmuir::KeyValueParser::parse (const QString & *line*)

Parse a string and assign a value to the correct parameter.

6.34.3.6 template<typename T > void Langmuir::KeyValueParser::registerVariable (const QString & *key,* T & *value,* Variable::VariableMode *mode* = 0) [private]

Register an allowed variable with the parser.

6.34.3.7 void Langmuir::KeyValueParser::save (const QString & *fileName* = "%stub.parm")

Write the parameters to a file in a "key=value" fashion.

6.34.3.8 QString Langmuir::KeyValueParser::toQString ( )

convert parameters to a QString

6.34.4 Friends And Related Function Documentation

6.34.4.1 std::ostream& operator<< (std::ostream & *stream,* const KeyValueParser & *keyValueParser*) [friend]

Write the parameters to a std::ostream in a "key=value" fashion.

6.34.4.2 QDebug operator<< (QDebug *dbg,* KeyValueParser & *keyValueParser*) [friend]

Write the parameters to QDebug in a "key=value" fashion.

6.34.5 Member Data Documentation

6.34.5.1 QStringList Langmuir::KeyValueParser::m_orderedNames [private]

A list of variable names in a specific order.

6.34.5.2 SimulationParameters Langmuir::KeyValueParser::m_parameters [private]

The simulation parameters.

6.34.5.3 QMap<QString,Variable*> Langmuir::KeyValueParser::m_variableMap [private]

A map between variable names and variables.

6.34.5.4 World& Langmuir::KeyValueParser::m_world [private]

Reference to World object.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/keyvalueparser.h

6.35 LangmuirViewer Class Reference

Widget to view Langmuir Simulation in real time.

`#include <langmuirviewer.h>`

Public Slots

- void setElectronPointMode (PointCloud::Mode mode)
  set the render mode for points
- void setDefectPointMode (PointCloud::Mode mode)
  set the render mode for points
- void setHolePointMode (PointCloud::Mode mode)
  set the render mode for points
- void setPointMode (PointCloud::Mode mode)
  set the render mode for points
- void setIterationsPrint (int value)
  set the value of iterations.print
- void toggleTrapsShown (bool on=true)
  turn on the traps texture
- void toggleOpenCL (bool on=true)
  turn OpenCL on and off
- void toggleCoulomb (bool on=true)
  turn Coulomb on and off
- void toggleCornerAxisIsVisible ()
  show/hide the corner axis
- void toggleGridIsVisible ()
  show/hide the grid
- void load (QString fileName)
  load a simulation
- void save (QString fileName)
  save a checkpoint file
- void resetCamera ()
  reset the camera to the default position
- void unload ()
  unload the current simulation
- void reset ()
  reset the current simulation
- void pause ()
  pause the simulation
- void play ()
  play the simulation
- void showParameters ()
  show parameters in a window
- void drawLightSource (GLenum light, float scale=1.0f) const
  draw an OpenGL light source
- void loadSettings (QString fileName)
  load settings from a file
- void saveSettings (QString fileName)
  save settings to a file
- void errorMessage (QString message)

show error message window

- void setSettings (QSettings &settings)

  set the properties of settings object
- void getSettings (QSettings &settings)

  get the properties of settings object
- void setTrapColor (QColor color)

  Change the trap color.
- void setStageColor2 (QColor color)

  Change the stage color2.
- void setBackgroundColor (QColor color)

  set background color
- void initTraps ()

  init trap texture
- void initStage ()

  init stage texture
- void generateIsoSurface (float value)

  generate isosurface
- void updateTrapMesh ()

  update trap mesh
- void resetSettings ()

  set default colors
- void setCanCalculateIsoSurface (bool enabled)

  set can calculate isosurface
- void setCheckerSize (float size)

  set size of checkers on stage

Signals

- void showMessage (const QString &message, int timeout=0)

  show a message in the status bar
- void clearMessage ()

  clear the status bar message
- void isUsingOpenCL (bool useOpenCL)

  signal if OpenCL is being used
- void isUsingCoulomb (bool useCoulomb)

  signal if Coulomb is being used
- void isAnimated (bool playing)

  signal if simulation is playing
- void isShowingTraps (bool shown)

  signal if traps are shown
- void iterationsPrintChanged (int value)

  signal that iterations.print changed
- void currentStepChanged (int value)

  signal that current.step changed
- void openGLInitFinished ()

  signal that OpenGL has been initialized
- void trapColorChanged (QColor color)

  signal that the trap color changed
- void stageColor2Changed (QColor color)

  signal that the stage color2 changed

- void backgroundColorChanged (QColor color)

  signal that the background color changed
- void canCalculateIsoSurface (bool able)

  signal that the isosurface can by calculated again
- void isoSurfaceProgress (int)

  show calculation progress
- void checkerSizeChanged (float)

  signal that the checker size changed

Public Member Functions

- LangmuirViewer (QWidget *parent=0)

  create the LangmuirViewer
- ~LangmuirViewer ()

  destroy the LangmuirViewer
- QMatrix4x4 & getModelViewProjectionMatrix ()

  obtain the model-view-projection matrix (QGLViewer camera)
- QMatrix4x4 & getProjectionMatrix ()

  obtain the projection matrix (QGLViewer camera)
- QMatrix4x4 & getOpenGLModelViewMatrix ()

  obtain the model-view matrix (OpenGL matrix stack)
- QMatrix4x4 & getOpenGLProjectionMatrix ()

  obtain the projection matrix (OpenGL matrix stack)
- Langmuir::Random & random ()

  get the random number generator
- bool okToCalculateIsoSurface ()

  its ok when not already calculating
- const QColor & trapColor () const

  get the trap color
- const QColor & stageColor2 () const

  get the stage color
- CornerAxis & cornerAxis ()

  get corner axis object
- PointCloud & electrons ()

  get electrons object
- PointCloud & defects ()

  get defects object
- PointCloud & holes ()

  get holes object
- PointCloud & traps ()

  get traps object
- Box & rightBox ()

  get right box object
- Box & leftBox ()

  get left box object
- Box & baseBox ()

  get base box object
- Box & trapBox ()

  get trap box object
- Box & stageBox ()

get stage box object

- Mesh & trapMesh ()

  get trap mesh object

- Light & light ()

  get light object

- Grid & grid ()

  get grid object

Protected Member Functions

- void updateElectronCloud ()

  update the electron point cloud

- void updateDefectCloud ()

  update the defect point cloud

- void updateTrapCloud ()

  update the defect point cloud

- void updateHoleCloud ()

  update the hole point cloud

- void initGeometry ()

  update the geometry using simulation parameters

- virtual void init ()

  setup OpenGL

- virtual void preDraw ()

  draw on the OpenGL widget before the main draw event

- virtual void draw ()

  draw on the OpenGL widget

- virtual void postDraw ()

  draw on the OpenGL widget after the main draw event

- virtual void animate ()

  change the state of the system before drawing

- virtual void help ()

  open the help widget

- virtual QString helpString () const

  get the help string

- void drawTraps (QImage &image, QColor bcolor, QColor fcolor)

  draw traps on image

- void drawChecker (QImage &image, QColor color1, QColor color2)

  draw checker pattern on stage

- void loadTexture (GLuint imageID, QImage &image)

  load texture to gpu memory

Protected Attributes

- GLuint m_textures [2]

  texture ids

- CornerAxis * m_cornerAxis

  axis that sits in the corner and doesnt change size

- PointCloud * m_electrons

  point cloud representing electrons

- PointCloud * m_defects

point cloud representing defects

- PointCloud * m_holes

  point cloud representing holes
- PointCloud * m_traps

  point cloud representing traps
- Box * m_trapBox

  trap box
- Box * m_baseBox

  base box
- Box * m_lBox

  box (left)
- Box * m_rBox

  box (right)
- Box * m_stageBox

  box (stage)
- double m_boxThickness

  box parameter
- QColor m_trapColor

  box parameter
- QColor m_stageColor2

  stage color
- Grid * m_grid

  grid that outlines sites
- Mesh * m_trapMesh

  trap mesh
- MarchingCubes::Isosurface * m_isoSurface

  isosurface
- Light * m_light0

  main light source
- Langmuir::Simulation * m_simulation

  the simulation manipulator
- Langmuir::Random m_random

  a random number generator instance
- Langmuir::World * m_world

  the simulation data
- float m_gridHalfX

  half of grid.x
- float m_gridHalfY

  half of grid.y
- float m_gridHalfZ

  half of grid.z
- float m_gridX

  grid.x
- float m_gridY

  grid.y
- float m_gridZ

  grid.z
- QErrorMessage * m_error

  error messages
- bool m_canCalculateIsoSurface

  can calculate isosurface

- float m_sceneRadius
  size of scene
- float m_stageExtend
  stage size
- float m_checkerSize
  size of checkers on stage

6.35.1 Detailed Description

Widget to view Langmuir Simulation in real time.

6.35.2 Constructor & Destructor Documentation

6.35.2.1 LangmuirViewer::LangmuirViewer (QWidget * *parent* = 0) [explicit]

create the LangmuirViewer

Parameters

parent	QObject this belongs to

6.35.2.2 LangmuirViewer::~LangmuirViewer ( )

destroy the LangmuirViewer

6.35.3 Member Function Documentation

6.35.3.1 virtual void LangmuirViewer::animate ( ) [protected], [virtual]

change the state of the system before drawing

6.35.3.2 void LangmuirViewer::backgroundColorChanged (QColor *color*) [signal]

signal that the background color changed

6.35.3.3 Box& LangmuirViewer::baseBox ( ) [inline]

get base box object

6.35.3.4 void LangmuirViewer::canCalculateIsoSurface (bool *able*) [signal]

signal that the isosurface can by calculated again

6.35.3.5 void LangmuirViewer::checkerSizeChanged (float) [signal]

signal that the checker size changed

6.35.3.6 void LangmuirViewer::clearMessage ( ) [signal]

clear the status bar message

6.35.3.7 CornerAxis& LangmuirViewer::cornerAxis ( ) [inline]

get corner axis object

6.35.3.8 void LangmuirViewer::currentStepChanged (int *value*) [signal]

signal that current.step changed

Parameters

value	value of current.step

6.35.3.9 PointCloud& LangmuirViewer::defects ( ) [inline]

get defects object

6.35.3.10 virtual void LangmuirViewer::draw ( ) [protected], [virtual]

draw on the OpenGL widget

6.35.3.11 void LangmuirViewer::drawChecker (QImage & *image,* QColor *color1,* QColor *color2*) [protected]

draw checker pattern on stage

Parameters

image	image to draw on

6.35.3.12 void LangmuirViewer::drawLightSource (GLenum *light,* float *scale* = 1.0f) const [slot]

draw an OpenGL light source

Parameters

light	
scale	

6.35.3.13 void LangmuirViewer::drawTraps (QImage & *image,* QColor *bcolor,* QColor *fcolor*) [protected]

draw traps on image

Parameters

image	image to draw draws on

6.35.3.14 PointCloud& LangmuirViewer::electrons ( ) [inline]

get electrons object

6.35.3.15 void LangmuirViewer::errorMessage (QString *message*) [slot]

show error message window

Parameters

message	error message

6.35.3.16 void LangmuirViewer::generateIsoSurface (float *value*) [slot]

generate isosurface

6.35.3.17 QMatrix4x4& LangmuirViewer::getModelViewProjectionMatrix ( )

obtain the model-view-projection matrix (QGLViewer camera)

6.35.3.18 QMatrix4x4& LangmuirViewer::getOpenGLModelViewMatrix ( )

obtain the model-view matrix (OpenGL matrix stack)

6.35.3.19 QMatrix4x4& LangmuirViewer::getOpenGLProjectionMatrix ( )

obtain the projection matrix (OpenGL matrix stack)

6.35.3.20 QMatrix4x4& LangmuirViewer::getProjectionMatrix ( )

obtain the projection matrix (QGLViewer camera)

6.35.3.21 void LangmuirViewer::getSettings (QSettings & *settings*) [slot]

get the properties of settings object

Parameters

settings	settings object

6.35.3.22 Grid& LangmuirViewer::grid ( ) [inline]

get grid object

6.35.3.23 virtual void LangmuirViewer::help ( ) [protected], [virtual]

open the help widget

6.35.3.24 virtual QString LangmuirViewer::helpString ( ) const [protected], [virtual]

get the help string

6.35.3.25 PointCloud& LangmuirViewer::holes ( ) [inline]

get holes object

6.35.3.26 virtual void LangmuirViewer::init ( ) [protected], [virtual]

setup OpenGL

6.35.3.27 void LangmuirViewer::initGeometry ( ) [protected]

update the geometry using simulation parameters

6.35.3.28 void LangmuirViewer::initStage ( ) [slot]

init stage texture

6.35.3.29 void LangmuirViewer::initTraps ( ) [slot]

init trap texture

6.35.3.30 void LangmuirViewer::isAnimated (bool *playing*) [signal]

signal if simulation is playing

Parameters

playing	true if playing

6.35.3.31 void LangmuirViewer::isoSurfaceProgress (int) [signal]

show calculation progress

6.35.3.32 void LangmuirViewer::isShowingTraps (bool *shown*) [signal]

signal if traps are shown

Parameters

shown	true if showing traps

6.35.3.33 void LangmuirViewer::isUsingCoulomb (bool *useCoulomb*) [signal]

signal if Coulomb is being used

Parameters

useCoulomb	true if using Coulomb

6.35.3.34 void LangmuirViewer::isUsingOpenCL (bool *useOpenCL*) [signal]

signal if OpenCL is being used

Parameters

useOpenCL	true if using OpenCL

6.35.3.35 void LangmuirViewer::iterationsPrintChanged (int *value*) [signal]

signal that iterations.print changed

Parameters

value	value of iterations.print

6.35.3.36 Box& LangmuirViewer::leftBox ( ) [inline]

get left box object

6.35.3.37 Light& LangmuirViewer::light ( ) [inline]

get light object

6.35.3.38 void LangmuirViewer::load (QString *fileName*) [slot]

load a simulation

Parameters

fileName	name of simulation input file

6.35.3.39 void LangmuirViewer::loadSettings (QString *fileName*) [slot]

load settings from a file

Parameters

fileName	name of settings file

6.35.3.40 void LangmuirViewer::loadTexture (GLuint *imageID*, QImage & *image*) [protected]

load texture to gpu memory

Parameters

image	texture

6.35.3.41 bool LangmuirViewer::okToCalculateIsoSurface ( ) [inline]

its ok when not already calculating

6.35.3.42 void LangmuirViewer::openGLInitFinished ( ) [signal]

signal that OpenGL has been initialized

6.35.3.43 void LangmuirViewer::pause ( ) [slot]

pause the simulation

6.35.3.44 void LangmuirViewer::play ( ) [slot]

play the simulation

6.35.3.45 virtual void LangmuirViewer::postDraw ( ) [protected], [virtual]

draw on the OpenGL widget after the main draw event

6.35.3.46 virtual void LangmuirViewer::preDraw ( ) [protected], [virtual]

draw on the OpenGL widget before the main draw event

6.35.3.47 Langmuir::Random& LangmuirViewer::random ( )

get the random number generator

6.35.3.48 void LangmuirViewer::reset ( ) [slot]

reset the current simulation

6.35.3.49 void LangmuirViewer::resetCamera ( ) [slot]

reset the camera to the default position

6.35.3.50 void LangmuirViewer::resetSettings ( ) [slot]

set default colors

6.35.3.51 Box& LangmuirViewer::rightBox ( ) [inline]

get right box object

6.35.3.52 void LangmuirViewer::save (QString *fileName*) [slot]

save a checkpoint file

Parameters

fileName	name of simulation checkpoint file

6.35.3.53 void LangmuirViewer::saveSettings (QString *fileName*) [slot]

save settings to a file

Parameters

fileName	name of settings file

6.35.3.54 void LangmuirViewer::setBackgroundColor (QColor *color*) [slot]

set background color

Parameters

color	color to set

6.35.3.55 void LangmuirViewer::setCanCalculateIsoSurface (bool *enabled*) [slot]

set can calculate isosurface

6.35.3.56 void LangmuirViewer::setCheckerSize (float *size*) [slot]

set size of checkers on stage

6.35.3.57 void LangmuirViewer::setDefectPointMode (PointCloud::Mode *mode*) [slot]

set the render mode for points

6.35.3.58 void LangmuirViewer::setElectronPointMode (PointCloud::Mode *mode*) [slot]

set the render mode for points

6.35.3.59 void LangmuirViewer::setHolePointMode (PointCloud::Mode *mode*) [slot]

set the render mode for points

6.35.3.60 void LangmuirViewer::setIterationsPrint (int *value*) [slot]

set the value of iterations.print

Parameters

value	value to set

6.35.3.61 void LangmuirViewer::setPointMode (PointCloud::Mode *mode*) [slot]

set the render mode for points

6.35.3.62 void LangmuirViewer::setSettings (QSettings & *settings*) [slot]

set the properties of settings object

Parameters

settings	settings object

6.35.3.63 void LangmuirViewer::setStageColor2 (QColor *color*) [slot]

Change the stage color2.

Parameters

color	color to set

6.35.3.64 void LangmuirViewer::setTrapColor (QColor *color*) [slot]

Change the trap color.

Parameters

color	color to set

6.35.3.65 void LangmuirViewer::showMessage (const QString & *message*, int *timeout* = 0) [signal]

show a message in the status bar

Parameters

message	string to display
timeout	message display time in ms

6.35.3.66 void LangmuirViewer::showParameters ( ) [slot]

show parameters in a window

6.35.3.67 Box& LangmuirViewer::stageBox ( ) [inline]

get stage box object

6.35.3.68 const QColor& LangmuirViewer::stageColor2 ( ) const [inline]

get the stage color

6.35.3.69 void LangmuirViewer::stageColor2Changed (QColor *color*) [signal]

signal that the stage color2 changed

6.35.3.70 void LangmuirViewer::toggleCornerAxisIsVisible ( ) [slot]

show/hide the corner axis

6.35.3.71 void LangmuirViewer::toggleCoulomb (bool *on* = true) [slot]

turn Coulomb on and off

Parameters

on	true if turning Coulomb on

6.35.3.72 void LangmuirViewer::toggleGridIsVisible ( ) [slot]

show/hide the grid

6.35.3.73 void LangmuirViewer::toggleOpenCL (bool *on* = true) [slot]

turn OpenCL on and off

Parameters

on	true if turning OpenCL on

6.35.3.74 void LangmuirViewer::toggleTrapsShown (bool *on* = true) [slot]

turn on the traps texture

Parameters

on	true is showing traps

6.35.3.75 Box& LangmuirViewer::trapBox ( ) [inline]

get trap box object

6.35.3.76 const QColor& LangmuirViewer::trapColor ( ) const [inline]

get the trap color

6.35.3.77 void LangmuirViewer::trapColorChanged (QColor *color*) [signal]

signal that the trap color changed

6.35.3.78 Mesh& LangmuirViewer::trapMesh ( ) [inline]

get trap mesh object

6.35.3.79 PointCloud& LangmuirViewer::traps ( ) [inline]

get traps object

6.35.3.80 void LangmuirViewer::unload ( ) [slot]

unload the current simulation

6.35.3.81 void LangmuirViewer::updateDefectCloud ( ) [protected]

update the defect point cloud

6.35.3.82 void LangmuirViewer::updateElectronCloud ( ) [protected]

update the electron point cloud

6.35.3.83 void LangmuirViewer::updateHoleCloud ( ) [protected]

update the hole point cloud

6.35.3.84 void LangmuirViewer::updateTrapCloud ( ) [protected]

update the defect point cloud

6.35.3.85 void LangmuirViewer::updateTrapMesh ( ) [slot]

update trap mesh

6.35.4 Member Data Documentation

6.35.4.1 Box* LangmuirViewer::m_baseBox [protected]

base box

6.35.4.2 double LangmuirViewer::m_boxThickness [protected]

box parameter

6.35.4.3 bool LangmuirViewer::m_canCalculateIsoSurface [protected]

can calculate isosurface

6.35.4.4 float LangmuirViewer::m_checkerSize [protected]

size of checkers on stage

6.35.4.5 CornerAxis* LangmuirViewer::m_cornerAxis [protected]

axis that sits in the corner and doesnt change size

6.35.4.6 PointCloud* LangmuirViewer::m_defects [protected]

point cloud representing defects

6.35.4.7 PointCloud* LangmuirViewer::m_electrons [protected]

point cloud representing electrons

6.35.4.8 QErrorMessage* LangmuirViewer::m_error [protected]

error messages

6.35.4.9 Grid* LangmuirViewer::m_grid [protected]

grid that outlines sites

6.35.4.10 float LangmuirViewer::m_gridHalfX [protected]

half of grid.x

6.35.4.11 float LangmuirViewer::m_gridHalfY [protected]

half of grid.y

6.35.4.12 float LangmuirViewer::m_gridHalfZ [protected]

half of grid.z

6.35.4.13 float LangmuirViewer::m_gridX [protected]

grid.x

6.35.4.14 float LangmuirViewer::m_gridY [protected]

grid.y

6.35.4.15 float LangmuirViewer::m_gridZ [protected]

grid.z

6.35.4.16 PointCloud* LangmuirViewer::m_holes [protected]

point cloud representing holes

6.35.4.17 MarchingCubes::Isosurface* LangmuirViewer::m_isoSurface [protected]

isosurface

6.35.4.18 Box* LangmuirViewer::m_lBox [protected]

box (left)

6.35.4.19 Light* LangmuirViewer::m_light0 [protected]

main light source

6.35.4.20 Langmuir::Random LangmuirViewer::m_random [protected]

a random number generator instance

6.35.4.21 Box* LangmuirViewer::m_rBox [protected]

box (right)

6.35.4.22 float LangmuirViewer::m_sceneRadius [protected]

size of scene

6.35.4.23 Langmuir::Simulation* LangmuirViewer::m_simulation [protected]

the simulation manipulator

6.35.4.24 Box* LangmuirViewer::m_stageBox [protected]

box (stage)

6.35.4.25 QColor LangmuirViewer::m_stageColor2 [protected]

stage color

6.35.4.26 float LangmuirViewer::m_stageExtend [protected]

stage size

6.35.4.27 GLuint LangmuirViewer::m_textures[2] [protected]

texture ids

6.35.4.28 Box* LangmuirViewer::m_trapBox [protected]

trap box

6.35.4.29 QColor LangmuirViewer::m_trapColor [protected]

box parameter

6.35.4.30 Mesh* LangmuirViewer::m_trapMesh [protected]

trap mesh

6.35.4.31 PointCloud* LangmuirViewer::m_traps [protected]

point cloud representing traps

6.35.4.32 Langmuir::World* LangmuirViewer::m_world [protected]

the simulation data

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

- /home/adam/opt/langmuir/src/langmuirView/include/langmuirviewer.h

6.36 Light Class Reference

A class to represent a light source.

`#include <light.h>`

Public Slots

- virtual void makeConnections ()

  make signal/slot connections
- void setPosition (float x, float y, float z, float w=0.0)

  set position of light
- void setPosition (QVector4D value)

  set position of light
- void setAColor (QColor color)

  set the ambient color
- void setDColor (QColor color)

  set the diffuse color
- void setSColor (QColor color)

  set the specular color
- void setLightID (GLuint lightID)

  set the OpenGL light ID
- void setEnabled (bool enabled)

  enabled or disable the light
- void toggle ()

  toggle the light between enabled/disabled
- void updatePosition ()

  use OpenGL commands
- void updateAColor ()

  use OpenGL commands
- void updateSColor ()

  use OpenGL commands
- void updateDColor ()

  use OpenGL commands

Signals

- void positionChanged (QVector4D position)

  signal that the position changed
- void aColorChanged (QColor color)

  signal that the ambient color of has changed
- void dColorChanged (QColor color)

  signal that the diffuse color of has changed
- void sColorChanged (QColor color)

  signal that the specular color of has changed
- void enabledChanged (bool enabled)

  signal that the light has been enabled/disabled
- void lightIDChanged (GLuint lightID)

  signal that the OpenGL light ID has changed

Public Member Functions

- Light (GLenum lightID, LangmuirViewer &viewer, QObject ∗parent=0)
- ∼Light ()
- const QColor & getAColor () const

  get ambient color
- const QColor & getDColor () const

  get diffuse color
- const QColor & getSColor () const

  get specular color
- GLenum getLightID () const

  get the OpenGL light id
- bool isEnabled () const

  find out if the light is enabled
- const QVector4D & getPosition () const

  get position of light

Protected Member Functions

- virtual void init ()

  initialize object
- virtual void draw ()

  perform OpenGL drawing operations

Protected Attributes

- QVector4D m_position

  location of light
- QColor m_acolor

  ambient color
- QColor m_dcolor

  diffuse color
- QColor m_scolor

  specular color
- GLenum m_lightID

  OpenGL light ID.
- bool m_enabled

  true if light is on

6.36.1 Detailed Description

A class to represent a light source.

6.36.2 Constructor & Destructor Documentation

6.36.2.1 Light::Light (GLenum *lightID,* LangmuirViewer & *viewer,* QObject ∗ *parent* = 0) [explicit]

6.36.2.2 Light::∼Light ( )

6.36.3 Member Function Documentation

6.36.3.1 void Light::aColorChanged (QColor *color*) [signal]

signal that the ambient color of has changed

Parameters

color	value of color

6.36.3.2 void Light::dColorChanged (QColor *color*) [signal]

signal that the diffuse color of has changed

Parameters

color	value of color

6.36.3.3 virtual void Light::draw ( ) [protected], [virtual]

perform OpenGL drawing operations

Reimplemented from SceneObject.

6.36.3.4 void Light::enabledChanged (bool *enabled*) [signal]

signal that the light has been enabled/disabled

Parameters

enabled	true if light has been enabled

6.36.3.5 const QColor& Light::getAColor ( ) const

get ambient color

6.36.3.6 const QColor& Light::getDColor ( ) const

get diffuse color

6.36.3.7 GLenum Light::getLightID ( ) const

get the OpenGL light id

6.36.3.8 const QVector4D& Light::getPosition ( ) const

get position of light

6.36.3.9 const QColor& Light::getSColor ( ) const

get specular color

6.36.3.10 virtual void Light::init ( ) [protected], [virtual]

initialize object

Reimplemented from SceneObject.

6.36.3.11 bool Light::isEnabled ( ) const

find out if the light is enabled

6.36.3.12 void Light::lightIDChanged (GLuint *lightID*) [signal]

signal that the OpenGL light ID has changed

Parameters

lightID	value of OpenGL light ID

6.36.3.13 virtual void Light::makeConnections ( ) [virtual], [slot]

make signal/slot connections

6.36.3.14 void Light::positionChanged (QVector4D *position*) [signal]

signal that the position changed

Parameters

position	value of position

6.36.3.15 void Light::sColorChanged (QColor *color*) [signal]

signal that the specular color of has changed

Parameters

color	value of color

6.36.3.16 void Light::setAColor (QColor *color*) [slot]

set the ambient color

Parameters

color	color to set

6.36.3.17 void Light::setDColor (QColor *color*) [slot]

set the diffuse color

Parameters

color	color to set

6.36.3.18 void Light::setEnabled (bool *enabled*) [slot]

enabled or disable the light

Parameters

enabled	true if the light is to be enabled

6.36.3.19 void Light::setLightID (GLuint *lightID*) [slot]

set the OpenGL light ID

Parameters

lightID	OpenGL light ID to use

6.36.3.20 void Light::setPosition (float *x*, float *y*, float *z*, float *w* = 0.0) [slot]

set position of light

Parameters

x	x-position
y	y-position
z	z-position
w	w-position

6.36.3.21 void Light::setPosition (QVector4D *value*) [slot]

set position of light

Parameters

value	position to set

6.36.3.22 void Light::setSColor (QColor *color*) [slot]

set the specular color

Parameters

color	color to set

6.36.3.23 void Light::toggle ( ) [slot]

toggle the light between enabled/disabled

6.36.3.24 void Light::updateAColor ( ) [slot]

use OpenGL commands

6.36.3.25 void Light::updateDColor ( ) [slot]

use OpenGL commands

6.36.3.26 void Light::updatePosition ( ) [slot]

use OpenGL commands

6.36.3.27 void Light::updateSColor ( ) [slot]

use OpenGL commands

6.36.4 Member Data Documentation

6.36.4.1 QColor Light::m_acolor [protected]

ambient color

6.36.4.2 QColor Light::m_dcolor [protected]

diffuse color

6.36.4.3 bool Light::m_enabled [protected]

true if light is on

6.36.4.4 GLenum Light::m_lightID [protected]

OpenGL light ID.

6.36.4.5 QVector4D Light::m_position [protected]

location of light

6.36.4.6 QColor Light::m_scolor [protected]

specular color

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

- /home/adam/opt/langmuir/src/langmuirView/include/light.h

6.37 Langmuir::Logger Class Reference

A class that organizes output.

`#include <writer.h>`

Public Member Functions

- Logger (World &world, QObject *parent=0)
  create Logger
- virtual void saveTrapImage (const QString &name="%stub-traps.png")
  save an image of trap sites as png

- virtual void saveHoleImage (const QString &name="%stub-%step-holes.png")

  save an image of holes (at the current step) as png
- virtual void saveElectronImage (const QString &name="%stub-%step-electrons.png")

  save an image of electrons (at the current step) as png
- virtual void saveCarriersImage (const QString &name="%stub-%step-carriers.png")

  *save an image of holes **and** electrons (at the current step) as png*
- virtual void saveDefectImage (const QString &name="%stub-defects.png")

  save an image of defects as png
- virtual void saveImage (const QString &name="%stub-%step-all.png")

  save an image of electrons, holes, defects, and traps (at current step) as png
- virtual void saveGridPotential (const QString &name="%stub.grid")

  output the grid potential as (x, y, z, v) to a file
- virtual void saveCoulombEnergy (const QString &name="%stub-%step.coulomb")

  *output the Coulomb potential as (x, y, z, v) to a file; **requires** the use of the **GPU***
- virtual void reportFluxStream ()

  output information about Sources and Drains (at the current step) to the main output file
- virtual void reportXYZStream ()

  output xyz information (at the current step) to the xyz file
- virtual void reportCarrier (ChargeAgent &charge)

  output carrier information (for example pathlength) to the carrier file
- virtual void reportExciton (ChargeAgent &charge1, ChargeAgent &charge2, bool recombined=false)

  output carrier information (for example pathlength) on two carriers at once to the exciton file
- virtual void initialize ()

  open the various output streams if they are turned on

Protected Attributes

- World & m_world

  reference to world
- XYZWriter * m_xyzWriter

  writer in charge of writing xyz files
- FluxWriter * m_fluxWriter

  writer in charge of writing source & drain information
- CarrierWriter * m_carrierWriter

  writer in charge of writing carrier information
- ExcitonWriter * m_excitonWriter

  writer in charge of writing multiple carrier's information (excitons)

6.37.1 Detailed Description

A class that organizes output.

Warning

You must manually call initialize() to open output streams

6.37.2 Constructor & Destructor Documentation

6.37.2.1 Langmuir::Logger::Logger (World & *world*, QObject * *parent* = 0)

create Logger

6.37.3 Member Function Documentation

6.37.3.1 virtual void Langmuir::Logger::initialize ( ) [virtual]

open the various output streams if they are turned on

6.37.3.2 virtual void Langmuir::Logger::reportCarrier (ChargeAgent & *charge*) [virtual]

output carrier information (for example pathlength) to the carrier file

6.37.3.3 virtual void Langmuir::Logger::reportExciton (ChargeAgent & *charge1,* ChargeAgent & *charge2,* bool *recombined* = false) [virtual]

output carrier information (for example pathlength) on two carriers at once to the exciton file

6.37.3.4 virtual void Langmuir::Logger::reportFluxStream ( ) [virtual]

output information about Sources and Drains (at the current step) to the main output file

6.37.3.5 virtual void Langmuir::Logger::reportXYZStream ( ) [virtual]

output xyz information (at the current step) to the xyz file

6.37.3.6 virtual void Langmuir::Logger::saveCarriersImage (const QString & *name* = "%stub-%step-carriers.png") [virtual]

save an image of holes **and** electrons (at the current step) as png

6.37.3.7 virtual void Langmuir::Logger::saveCoulombEnergy (const QString & *name* = "%stub-%step.coulomb") [virtual]

output the Coulomb potential as (x, y, z, v) to a file; **requires** the use of the **GPU**

6.37.3.8 virtual void Langmuir::Logger::saveDefectImage (const QString & *name* = "%stub-defects.png") [virtual]

save an image of defects as png

6.37.3.9 virtual void Langmuir::Logger::saveElectronImage (const QString & *name* = "%stub-%step-electrons.png") [virtual]

save an image of electrons (at the current step) as png

6.37.3.10 virtual void Langmuir::Logger::saveGridPotential (const QString & *name* = "%stub.grid") [virtual]

output the grid potential as (x, y, z, v) to a file

6.37.3.11 virtual void Langmuir::Logger::saveHoleImage (const QString & *name* = "%stub-%step-holes.png") [virtual]

save an image of holes (at the current step) as png

6.37.3.12 virtual void Langmuir::Logger::saveImage (const QString & *name* = "%stub-%step-all.png") [virtual]

save an image of electrons, holes, defects, and traps (at current step) as png

6.37.3.13 virtual void Langmuir::Logger::saveTrapImage (const QString & *name* = "%stub-traps.png") [virtual]

save an image of trap sites as png

6.37.4 Member Data Documentation

6.37.4.1 CarrierWriter* Langmuir::Logger::m_carrierWriter [protected]

writer in charge of writing carrier information

6.37.4.2 ExcitonWriter* Langmuir::Logger::m_excitonWriter [protected]

writer in charge of writing multiple carrier's information (excitons)

6.37.4.3 FluxWriter* Langmuir::Logger::m_fluxWriter [protected]

writer in charge of writing source & drain information

6.37.4.4 World& Langmuir::Logger::m_world [protected]

reference to world

6.37.4.5 XYZWriter* Langmuir::Logger::m_xyzWriter [protected]

writer in charge of writing xyz files

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

- /home/adam/opt/langmuir/src/langmuirCore/include/writer.h

6.38 MainWindow Class Reference

A window with an OpenGL widget.

`#include <mainwindow.h>`

Public Slots

- void on_actionScreenshot_triggered ()
  take a screen shot
- void on_actionOpen_triggered ()
  open an input file
- void on_actionSave_triggered ()
  save checkpoint file
- void on_actionResetSettings_triggered ()

reset settings (looks for config.ini)

- void on_actionLoadSettings_triggered ()

  load settings file
- void on_actionSaveSettings_triggered ()

  save settings file
- void on_actionPoints_triggered ()

  open point dialog
- void on_actionIsoSurface_triggered ()

  open isosurface dialog
- void on_actionChecker_triggered ()

  open checker dialog
- void setStopEnabled (bool enabled)

  enabled or disable stop and play buttons
- void closeEvent (QCloseEvent *event)

  clean up before closing
- void initAfter ()

  setup some connections that require OpenGL to be initialized first
- void updateSpinBox (int value)

  update the iterations.print spin box

Public Member Functions

- MainWindow (const QString &inputFile="", QWidget *parent=0)
- ~MainWindow ()

Private Member Functions

- void setIcon (QAction *action, QString themeIcon, QStyle::StandardPixmap standardPixmap)

  set user interface icon
- void init ()

  setup

Private Attributes

- Ui::MainWindow * ui

  main window user interface
- LangmuirViewer * m_viewer

  OpenGL widget.
- PointDialog * m_pointdialog

  dialog to set point parameters
- IsoSurfaceDialog * m_isosurfacedialog

  dialog to create isosurface
- QDir m_currentDir

  current directory for open/save actions

6.38.1 Detailed Description

A window with an OpenGL widget.

6.38.2 Constructor & Destructor Documentation

6.38.2.1 MainWindow::MainWindow (const QString & *inputFile* = " ", QWidget * *parent* = 0) [explicit]

6.38.2.2 MainWindow::~MainWindow ( )

6.38.3 Member Function Documentation

6.38.3.1 void MainWindow::closeEvent (QCloseEvent * *event*) [slot]

clean up before closing

6.38.3.2 void MainWindow::init ( ) [private]

setup

6.38.3.3 void MainWindow::initAfter ( ) [slot]

setup some connections that require OpenGL to be initialized first

6.38.3.4 void MainWindow::on_actionChecker_triggered ( ) [slot]

open checker dialog

6.38.3.5 void MainWindow::on_actionIsoSurface_triggered ( ) [slot]

open isosurface dialog

6.38.3.6 void MainWindow::on_actionLoadSettings_triggered ( ) [slot]

load settings file

6.38.3.7 void MainWindow::on_actionOpen_triggered ( ) [slot]

open an input file

6.38.3.8 void MainWindow::on_actionPoints_triggered ( ) [slot]

open point dialog

6.38.3.9 void MainWindow::on_actionResetSettings_triggered ( ) [slot]

reset settings (looks for config.ini)

6.38.3.10 void MainWindow::on_actionSave_triggered ( ) [slot]

save checkpoint file

6.38.3.11 void MainWindow::on_actionSaveSettings_triggered ( ) [slot]

save settings file

6.38.3.12 void MainWindow::on_actionScreenshot_triggered ( ) [slot]

take a screen shot

6.38.3.13 void MainWindow::setIcon (QAction * *action,* QString *themeIcon,* QStyle::StandardPixmap *standardPixmap*) [private]

set user interface icon

6.38.3.14 void MainWindow::setStopEnabled (bool *enabled*) [slot]

enabled or disable stop and play buttons

6.38.3.15 void MainWindow::updateSpinBox (int *value*) [slot]

update the iterations.print spin box

6.38.4 Member Data Documentation

6.38.4.1 QDir MainWindow::m_currentDir [private]

current directory for open/save actions

6.38.4.2 IsoSurfaceDialog* MainWindow::m_isosurfacedialog [private]

dialog to create isosurface

6.38.4.3 PointDialog* MainWindow::m_pointdialog [private]

dialog to set point parameters

6.38.4.4 LangmuirViewer* MainWindow::m_viewer [private]

OpenGL widget.

6.38.4.5 Ui::MainWindow* MainWindow::ui [private]

main window user interface

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

- /home/adam/opt/langmuir/src/langmuirView/include/mainwindow.h

6.39 Langmuir::MainWindow Class Reference

`#include <gridview.h>`

Public Slots

- void setConnections ()

Public Member Functions

- MainWindow (QString input)

Public Attributes

- GridViewGL * glWidget
- Navigator * navigator
- SceneOptions * sceneOptions
- Controls * controls

6.39.1 Constructor & Destructor Documentation

6.39.1.1 Langmuir::MainWindow::MainWindow (QString *input*)

6.39.2 Member Function Documentation

6.39.2.1 void Langmuir::MainWindow::setConnections ( ) [slot]

6.39.3 Member Data Documentation

6.39.3.1 Controls* Langmuir::MainWindow::controls

6.39.3.2 GridViewGL* Langmuir::MainWindow::glWidget

6.39.3.3 Navigator* Langmuir::MainWindow::navigator

6.39.3.4 SceneOptions* Langmuir::MainWindow::sceneOptions

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.40 Mesh Class Reference

A class to represent a mesh.

`#include <mesh.h>`

Public Types

- enum Mode {
  Single = 1, SingleAlpha = 2, Double = 3, DoubleAlpha = 4,
  Shader1 = 5, Shader2 = 6 }

  The rendering mode for the cloud.

Public Slots

- virtual void makeConnections ()

  make signal/slot connections
- void setColorA (QColor color)

  set color A
- void setColorB (QColor color)

  set color B
- void setMesh (const QVector< float > &vertices, const QVector< float > &normals, const QVector< unsigned int > &indices)

  set up the mesh on the GPU
- void setMode (Mesh::Mode mode)

  set the mode
- void clear ()

  clear GPU buffers
- static QString modeToQString (Mode mode)

  convert Mode to string
- static Mode QStringToMode (QString string)

  convert string to Mode enum

Signals

- void colorAChanged (QColor color)

  signal that color A of has changed
- void colorBChanged (QColor color)

  signal that color B of has changed
- void meshChanged ()

  signal that the mesh has changed
- void modeChanged (Mesh::Mode mode)

  signal that the render mode has changed
- void modeChanged (QString mode)

  signal that the render mode has changed

Public Member Functions

- Mesh (LangmuirViewer &viewer, QObject *parent=0)

  create the Mesh
- ~Mesh ()

  destroy the Mesh
- const QColor & getColorA () const

  get color A
- const QColor & getColorB () const

  get color B
- Mode getMode () const

  get render mode

Protected Member Functions

- virtual void init ()
  initialize object
- virtual void draw ()
  perform OpenGL drawing operations
- void initShaders ()
  load the shaders
- void drawSingle ()
  render function
- void drawSingleAlpha ()
  render function
- void drawDouble ()
  render function
- void drawDoubleAlpha ()
  render function
- void drawShader1 ()
  render function
- void drawShader2 ()
  render function

Protected Attributes

- QOpenGLShaderProgram m_shader1
  tesselation shader
- QOpenGLShaderProgram m_shader2
  tesselation shader
- QOpenGLBuffer * m_verticesVBO
  vertices buffer
- QOpenGLBuffer * m_normalsVBO
  normals buffer
- QOpenGLBuffer * m_indexVBO
  index buffer CW
- QColor m_colorA
  color of side A
- QColor m_colorB
  color of side B
- unsigned int m_numVertices
  *number of vertices (3 * number of points)*
- unsigned int m_numIndices
  index count
- Mode m_mode
  rendering mode
- bool m_shader1OK
  shader1 ok to use
- bool m_shader2OK
  shader2 ok to use

6.40.1 Detailed Description

A class to represent a mesh.

6.40.2 Member Enumeration Documentation

6.40.2.1 enum Mesh::Mode

The rendering mode for the cloud.

Enumerator

Single

SingleAlpha render mesh using single color

Double render mesh using single color with alpha blending

DoubleAlpha render mesh using two colors

Shader1 render mesh using two colors with alpha blending

Shader2 render mesh with shader1

6.40.3 Constructor & Destructor Documentation

6.40.3.1 Mesh::Mesh (LangmuirViewer & *viewer,* QObject * *parent* = 0) [explicit]

create the Mesh

Parameters

viewer	the viewer
parent	QObject this belongs to

6.40.3.2 Mesh::~Mesh ( )

destroy the Mesh

6.40.4 Member Function Documentation

6.40.4.1 void Mesh::clear ( ) [slot]

clear GPU buffers

6.40.4.2 void Mesh::colorAChanged (QColor *color*) [signal]

signal that color A of has changed

Parameters

color	value of color

6.40.4.3 void Mesh::colorBChanged (QColor *color*) [signal]

signal that color B of has changed

Parameters

color	value of color

6.40.4.4 virtual void Mesh::draw ( ) [protected], [virtual]

perform OpenGL drawing operations

Reimplemented from SceneObject.

6.40.4.5 void Mesh::drawDouble ( ) [protected]

render function

6.40.4.6 void Mesh::drawDoubleAlpha ( ) [protected]

render function

6.40.4.7 void Mesh::drawShader1 ( ) [protected]

render function

6.40.4.8 void Mesh::drawShader2 ( ) [protected]

render function

6.40.4.9 void Mesh::drawSingle ( ) [protected]

render function

6.40.4.10 void Mesh::drawSingleAlpha ( ) [protected]

render function

6.40.4.11 const QColor& Mesh::getColorA ( ) const

get color A

6.40.4.12 const QColor& Mesh::getColorB ( ) const

get color B

6.40.4.13 Mode Mesh::getMode ( ) const

get render mode

6.40.4.14 virtual void Mesh::init ( ) [protected], [virtual]

initialize object

Reimplemented from SceneObject.

6.40.4.15 void Mesh::initShaders ( ) [protected]

load the shaders

6.40.4.16 virtual void Mesh::makeConnections ( ) [virtual], [slot]

make signal/slot connections

6.40.4.17 void Mesh::meshChanged ( ) [signal]

signal that the mesh has changed

6.40.4.18 void Mesh::modeChanged (Mesh::Mode *mode*) [signal]

signal that the render mode has changed

Parameters

mode	value of rendering mode

6.40.4.19 void Mesh::modeChanged (QString *mode*) [signal]

signal that the render mode has changed

Parameters

mode	value of rendering mode

6.40.4.20 static QString Mesh::modeToQString (Mode *mode*) [static], [slot]

convert Mode to string

Parameters

mode	mode enum

6.40.4.21 static Mode Mesh::QStringToMode (QString *string*) [static], [slot]

convert string to Mode enum

Parameters

string	mode string

6.40.4.22 void Mesh::setColorA (QColor *color*) [slot]

set color A

Parameters

color	color to set

6.40.4.23 void Mesh::setColorB (QColor *color*) [slot]

set color B

Parameters

color	color to set

6.40.4.24 void Mesh::setMesh (const QVector< float > & *vertices*, const QVector< float > & *normals*, const QVector< unsigned int > & *indices*) [slot]

set up the mesh on the GPU

Parameters

vertices	vertex buffer
normals	normal buffer
indices	index buffer

6.40.4.25 void Mesh::setMode (Mesh::Mode *mode*) [slot]

set the mode

Parameters

mode	mode to set

6.40.5 Member Data Documentation

6.40.5.1 QColor Mesh::m_colorA [protected]

color of side A

6.40.5.2 QColor Mesh::m_colorB [protected]

color of side B

6.40.5.3 QOpenGLBuffer* Mesh::m_indexVBO [protected]

index buffer CW

6.40.5.4 Mode Mesh::m_mode [protected]

rendering mode

6.40.5.5 QOpenGLBuffer* Mesh::m_normalsVBO [protected]

normals buffer

6.40.5.6 unsigned int Mesh::m_numIndices [protected]

index count

6.40.5.7 unsigned int Mesh::m_numVertices [protected]

number of vertices (3 * number of points)

6.40.5.8 QOpenGLShaderProgram Mesh::m_shader1 [protected]

tesselation shader

6.40.5.9 bool Mesh::m_shader1OK [protected]

shader1 ok to use

6.40.5.10 QOpenGLShaderProgram Mesh::m_shader2 [protected]

tesselation shader

6.40.5.11 bool Mesh::m_shader2OK [protected]

shader2 ok to use

6.40.5.12 QOpenGLBuffer* Mesh::m_verticesVBO [protected]

vertices buffer

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

- /home/adam/opt/langmuir/src/langmuirView/include/mesh.h

6.41 Langmuir::Navigator Class Reference

`#include <gridview.h>`

Public Member Functions

- Navigator (QWidget *parent)

Public Attributes

- QGridLayout * layout
- QList< QLabel * > labels
- QList< DSpinBox * > spinBoxes
- QList< Button * > buttons

6.41.1 Constructor & Destructor Documentation

6.41.1.1 Langmuir::Navigator::Navigator (QWidget * *parent*)

6.41.2 Member Data Documentation

6.41.2.1 QList< Button* > Langmuir::Navigator::buttons

6.41.2.2 QList< QLabel* > Langmuir::Navigator::labels

6.41.2.3 QGridLayout* Langmuir::Navigator::layout

6.41.2.4 QList< DSpinBox* > Langmuir::Navigator::spinBoxes

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.42 Langmuir::NodeFileParser Class Reference

`#include <nodefileparser.h>`

Public Member Functions

- NodeFileParser (const QString &nodefile="", const QString &gpufile="", QObject *parent=0)
  create NodeFileParser
- void setPaths (const QString &nodefile="", const QString &gpufile="")
  aquire the paths of the GPUFILE and the NODEFILE and parse them
- void setHostName (const QString &hostName)
  set the name of this CPU
- void setHostName ()
  set the name of this CPU using boost::asio::ip::hostname()
- void setDefault ()
  set the default based on QThreadPool and hostname
- void createNode (const QString &name, int cores=0, QList< int > gpus=QList< int >())
  add cpu to records
- void clear ()
  clear the records
- int numProc ()
  get the number of processes on all hosts
- int numProc (const QString &name)
  get the number of processes on host
- const QMap< QString, int > & procs ()
  get the number of processes on all hosts
- int numGPUS ()
  get the number of gpus on all hosts
- int numGPUs (const QString &name)
  get the number of gpus on host
- int GPUid (const QString &name, int i)
  get the ith gpu id on host
- const QList< int > & gpus (const QString &name)

get the number of gpus on host

- int numCPUs ()

  get the number of hosts

- const QStringList & cpus ()

  get the hostnames

- const QString & hostName ()

  get the hostname of this cpu

Private Member Functions

- bool parse (QString &filename, bool ignoreCores=false, bool ignoreGPUs=true)

Private Attributes

- QStringList m_names

  list of cpu names

- QMap< QString, int > m_cores

  list of core counts per cpu

- QMap< QString, QList< int > > m_gpus

  list of gpu ids per cpu

- QString m_nodefile

  path to NODEFILE

- QString m_gpufile

  path to GPUFILE

- QString m_hostName

  hostname of this computer

Friends

- QDebug operator<< (QDebug dbg, const NodeFileParser &nfp)

  operator overload for QDebug

6.42.1 Detailed Description

A class to parse the PBS_NODEFILE and PBS_GPUFILE

6.42.2 Constructor & Destructor Documentation

6.42.2.1 Langmuir::NodeFileParser::NodeFileParser (const QString & *nodefile* = " ", const QString & *gpufile* = " ", QObject * *parent* = 0) [explicit]

create NodeFileParser

Parameters

nodefile	path to NODEFILE
gpufile	path to GPUFILE

6.42.3 Member Function Documentation

6.42.3.1 void Langmuir::NodeFileParser::clear ( )

clear the records

6.42.3.2 const QStringList& Langmuir::NodeFileParser::cpus ( )

get the hostnames

6.42.3.3 void Langmuir::NodeFileParser::createNode (const QString & *name,* int *cores* = 0, QList< int > *gpus* = QList< int >())

add cpu to records

Parameters

name	name of cpu

6.42.3.4 int Langmuir::NodeFileParser::GPUid (const QString & *name,* int *i*)

get the ith gpu id on host

Parameters

name	hostname
i	index

6.42.3.5 const QList<int>& Langmuir::NodeFileParser::gpus (const QString & *name*)

get the number of gpus on host

Parameters

name	hostname

6.42.3.6 const QString& Langmuir::NodeFileParser::hostName ( )

get the hostname of this cpu

6.42.3.7 int Langmuir::NodeFileParser::numCPUs ( )

get the number of hosts

6.42.3.8 int Langmuir::NodeFileParser::numGPUS ( )

get the number of gpus on all hosts

6.42.3.9 int Langmuir::NodeFileParser::numGPUs (const QString & *name*)

get the number of gpus on host

Parameters

name	hostname

6.42.3.10 int Langmuir::NodeFileParser::numProc ( )

get the number of processes on all hosts

6.42.3.11 int Langmuir::NodeFileParser::numProc (const QString & *name*)

get the number of processes on host

Parameters

name	hostname

6.42.3.12 bool Langmuir::NodeFileParser::parse (QString & *filename*, bool *ignoreCores* = `false`, bool *ignoreGPUs* = `true`) [private]

parse NODEFILE or GPUFILE

Parameters

filename	name of file to parse
ignoreCores	do not parse core information
ignoreGPUs	do not parse gpu information

Returns

true if success

6.42.3.13 const QMap<QString,int>& Langmuir::NodeFileParser::procs ( )

get the number of processes on all hosts

6.42.3.14 void Langmuir::NodeFileParser::setDefault ( )

set the default based on QThreadPool and hostname

6.42.3.15 void Langmuir::NodeFileParser::setHostName (const QString & *hostName*)

set the name of this CPU

Parameters

hostName	host name string

6.42.3.16 void Langmuir::NodeFileParser::setHostName ( )

set the name of this CPU using boost::asio::ip::hostname()

6.42.3.17 void Langmuir::NodeFileParser::setPaths (const QString & *nodefile* = " ", const QString & *gpufile* = " ")

aquire the paths of the GPUFILE and the NODEFILE and parse them

Parameters

nodefile	path to NODEFILE
gpufile	path to GPUFILE

If nodefile is empty the enviroment variable PBS_NODEFILE is used. If gpufile is empty, the enviroment variable PBS_GPUFILE is used. If both paths remain empty, then setDefault() is used.

6.42.4 Friends And Related Function Documentation

6.42.4.1 QDebug operator<<(QDebug *dbg,* const NodeFileParser & *nfp*) [friend]

operator overload for QDebug

6.42.5 Member Data Documentation

6.42.5.1 QMap<QString,int> Langmuir::NodeFileParser::m_cores [private]

list of core counts per cpu

6.42.5.2 QString Langmuir::NodeFileParser::m_gpufile [private]

path to GPUFILE

6.42.5.3 QMap<QString,QList<int> > Langmuir::NodeFileParser::m_gpus [private]

list of gpu ids per cpu

6.42.5.4 QString Langmuir::NodeFileParser::m_hostName [private]

hostname of this computer

6.42.5.5 QStringList Langmuir::NodeFileParser::m_names [private]

list of cpu names

6.42.5.6 QString Langmuir::NodeFileParser::m_nodefile [private]

path to NODEFILE

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

- /home/adam/opt/langmuir/src/langmuirCore/include/nodefileparser.h

6.43 Langmuir::OpenCLHelper Class Reference

A Class to run OpenCL calculations.

`#include <openclhelper.h>`

Public Member Functions

- OpenCLHelper (World &world, QObject *parent=0)

  *Create **THE** OpenCLHelper; don't make more than one.*
- void initializeOpenCL (int gpuID=-1)

  Perform the tedious boilerplate code to initialize OpenCL.
- void launchCoulombKernel1 ()

  *Kernel1 calculates the coulomb potential at **every** site.*
- void launchCoulombKernel2 ()

  Kernel2 calculates the coulomb potential at current and future sites only.
- void launchGaussKernel1 ()

  *Kernel1 calculates the coulomb potential with erf at **every** site.*
- void launchGaussKernel2 ()

  Kernel2 calculates the coulomb potential with erf at current and future sites only.
- void copySiteAndChargeToHostVector (int index, int site, int charge=-1)

  Does exactly what it says (host means the memory on the CPU)
- double getOutputHost (int index) const

  Get the result stored in host memory (for current site)
- double getOutputHostFuture (int index) const

  Get the result stored in host memory (for future site)
- void compareHostAndDeviceForAllCarriers ()

  Compare GPU and CPU results.
- bool toggleOpenCL (bool on)

  Turn on/off OpenCL in a smart-way.

Private Attributes

- World & m_world

  Reference to World object.

6.43.1 Detailed Description

A Class to run OpenCL calculations.

6.43.2 Constructor & Destructor Documentation

6.43.2.1 Langmuir::OpenCLHelper::OpenCLHelper (World & *world*, QObject * *parent* = 0)

Create **THE** OpenCLHelper; don't make more than one.

Parameters

world	reference to World Object
parent	QObject this belongs to

Warning

initializeOpenCL() must be called seperately

6.43.3 Member Function Documentation

6.43.3.1 void Langmuir::OpenClHelper::compareHostAndDeviceForAllCarriers ( )

Compare GPU and CPU results.

6.43.3.2 void Langmuir::OpenClHelper::copySiteAndChargeToHostVector (int *index*, int *site*, int *charge* = −1)

Does exactly what it says (host means the memory on the CPU)

Parameters

index	position in host vectors
site	serial site-id
charge	charge of carrier

6.43.3.3 double Langmuir::OpenClHelper::getOutputHost (int *index*) const

Get the result stored in host memory (for current site)

Parameters

index	position in host vectors

6.43.3.4 double Langmuir::OpenClHelper::getOutputHostFuture (int *index*) const

Get the result stored in host memory (for future site)

Parameters

index	position in host vectors

There is a fixed offset in the host memory between the current and future site results

6.43.3.5 void Langmuir::OpenClHelper::initializeOpenCL (int *gpuID* = −1)

Perform the tedious boilerplate code to initialize OpenCL.

6.43.3.6 void Langmuir::OpenClHelper::launchCoulombKernel1 ( )

Kernel1 calculates the coulomb potential at **every** site.

This is extremely expensive on the CPU, it would take forever. The GPU will do it in a few seconds. Luckily, the only reason to ever call this kernel is if we want to save a snapshot of the coulomb potential - something that is not needed during a normal simulation.

6.43.3.7 void Langmuir::OpenClHelper::launchCoulombKernel2 ( )

Kernel2 calculates the coulomb potential at current and future sites only.

6.43.3.8 void Langmuir::OpenClHelper::launchGaussKernel1 ( )

Kernel1 calculates the coulomb potential with erf at **every** site.

6.43.3.9 void Langmuir::OpenClHelper::launchGaussKernel2 ( )

Kernel2 calculates the coulomb potential with erf at current and future sites only.

6.43.3.10 bool Langmuir::OpenClHelper::toggleOpenCL (bool *on*)

Turn on/off OpenCL in a smart-way.

Parameters

on	True if on

Returns

The on/off status

For example, don't allow one to turn OpenCL on if OpenCL can't be used on this platform.

6.43.4 Member Data Documentation

6.43.4.1 World& Langmuir::OpenClHelper::m_world [private]

Reference to World object.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/openclhelper.h

6.44 Langmuir::OutputInfo Class Reference

A class to generate file names using the SimulationParameters.

`#include <output.h>`

Public Member Functions

- OutputInfo (const QString &name, const SimulationParameters *par=0)

  Generate file name according to SimulationParameters.

6.44.1 Detailed Description

A class to generate file names using the SimulationParameters.

6.44.2 Constructor & Destructor Documentation

6.44.2.1 Langmuir::OutputInfo::OutputInfo (const QString & *name*, const SimulationParameters * *par* = 0)

Generate file name according to SimulationParameters.

The constructor makes useful substitutions into the passed name (deatiled below) as well as making sure the name generated is valid (according to the passed SimulationParameters). If the directory of the passed name doesn't exist, it will be created.

Parameters

name	the file name desired. The following substitutions can be made: • **"%stub"**, substitutes in SimulationParameters::outputStub • **"%step"**, substitutes in SimulationParameters::currentStep
par	pointer to a SimulationParameters object • if 0 or NULL, then all substitutions become empty strings

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

- /home/adam/opt/langmuir/src/langmuirCore/include/output.h

6.45 Langmuir::OutputStream Class Reference

A class to combine QFile, QTextStream and OutputInfo (QFileInfo).

`#include <output.h>`

Public Member Functions

- OutputStream (const QString &name, const SimulationParameters *par=0, QObject *parent=0)
  Setup the QTextStream, QFile, and OutputInfo.
- ~OutputStream ()
  Flush the stream and close the file.
- const OutputInfo & info ()
  Get the info object to get things like file name and path.
- const QFile & file ()
  Get the file object, though you probably have no need for it.

Private Attributes

- OutputInfo m_info
  OutputInfo object that generated file name.
- QFile m_file
  QFile object, the device of this QTextStream.

6.45.1 Detailed Description

A class to combine QFile, QTextStream and OutputInfo (QFileInfo).

Only for used for output. Derived from QObject so destruction ensures streams are flushed and files are closed.

6.45.2 Constructor & Destructor Documentation

6.45.2.1 Langmuir::OutputStream::OutputStream (const QString & *name*, const SimulationParameters * *par* = 0, QObject * *parent* = 0)

Setup the QTextStream, QFile, and OutputInfo.

The parameters are the same as OutputInfo. Opens the file as QIODevice::Text|QIODevice::WriteOnly. Will open with QIODevice::Append if Outout::Options::AppendMode is given.

See also

OutputInfo::OutputInfo

6.45.2.2 Langmuir::OutputStream::~OutputStream ( )

Flush the stream and close the file.

6.45.3 Member Function Documentation

6.45.3.1 const QFile& Langmuir::OutputStream::file ( )

Get the file object, though you probably have no need for it.

6.45.3.2 const OutputInfo& Langmuir::OutputStream::info ( )

Get the info object to get things like file name and path.

6.45.4 Member Data Documentation

6.45.4.1 QFile Langmuir::OutputStream::m_file [private]

QFile object, the device of this QTextStream.

6.45.4.2 OutputInfo Langmuir::OutputStream::m_info [private]

OutputInfo object that generated file name.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/output.h

6.46 Langmuir::PointArray Class Reference

`#include <gridview.h>`

Public Slots

- void setSpheres (int checkState)
- void setPointSize (double pointSize)

Signals

- void pointSizeChanged (double pointSize)

Public Member Functions

- PointArray (QObject *parent, QVector< float > &xyz)
- ~PointArray ()
- void draw (int size, int height, float fov)
- void update (QVector< float > &xyz, int size)

Private Attributes

- QGLShaderProgram program
- QGLBuffer vBuffer
- float pointSize
- bool spheres

Additional Inherited Members

6.46.1 Constructor & Destructor Documentation

6.46.1.1 Langmuir::PointArray::PointArray (QObject * *parent,* QVector< float > & *xyz*)

6.46.1.2 Langmuir::PointArray::~PointArray ( )

6.46.2 Member Function Documentation

6.46.2.1 void Langmuir::PointArray::draw (int *size,* int *height,* float *fov*)

6.46.2.2 void Langmuir::PointArray::pointSizeChanged (double *pointSize*) [signal]

6.46.2.3 void Langmuir::PointArray::setPointSize (double *pointSize*) [slot]

6.46.2.4 void Langmuir::PointArray::setSpheres (int *checkState*) [slot]

6.46.2.5 void Langmuir::PointArray::update (QVector< float > & *xyz,* int *size*)

6.46.3 Member Data Documentation

6.46.3.1 float Langmuir::PointArray::pointSize [private]

6.46.3.2 QGLShaderProgram Langmuir::PointArray::program [private]

6.46.3.3 bool Langmuir::PointArray::spheres [private]

6.46.3.4 QGLBuffer Langmuir::PointArray::vBuffer [private]

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.47 PointCloud Class Reference

A class to represent a point cloud.

`#include <pointcloud.h>`

Public Types

- enum Mode { Points = 1, Squares = 2, Cubes = 3 }

  The rendering mode for the cloud.

Public Slots

- virtual void makeConnections ()

  make signal/slot connections
- void setMaxPoints (unsigned int value)

  change the maximum number of points allowed
- void setMaxRender (unsigned int value)

  change the maximum number of points to render
- void setPointSize (float value)

  set the point size
- void setColor (QColor color)

  set the color
- void setMode (Mode mode)

  set the mode
- void updateVBO ()

  update the GPU memory using CPU vertices
- static QString modeToQString (Mode mode)

  convert Mode to string
- static Mode QStringToMode (QString string)

  convert string to Mode enum

Signals

- void maxPointsChanged (unsigned int value)

  signal that the maximum number of points has changed
- void maxRenderChanged (unsigned int value)

  signal that the maximum number of points rendered has changed
- void pointSizeChanged (float value)

  signal that the point size has changed
- void colorChanged (QColor color)

  signal that the color of has changed
- void modeChanged (PointCloud::Mode mode)

  signal that the render mode has changed
- void modeChanged (QString mode)

  signal that the render mode has changed

Public Member Functions

- PointCloud (LangmuirViewer &viewer, QObject *parent=0)

  create the PointCloud
- ~PointCloud ()

  destroy the PointCloud
- const QColor & getColor () const

  get color
- float getPointSize () const

  get x length
- Mode getMode () const

  get render mode
- QVector< float > & vertices ()

  get the list of vertices (CPU memory)

- unsigned int getMaxPoints ()
  get the maximum number of points that can be rendered
- unsigned int getMaxRender ()
  get the number of points currently being rendered

Protected Member Functions

- virtual void init ()
  initialize object
- virtual void draw ()
  perform OpenGL drawing operations
- void initShaders ()
  load the shaders
- void drawFallback ()
  render function
- void drawPoints ()
  render function
- void drawSquares ()
  render function
- void drawCubes ()
  render function

Protected Attributes

- QOpenGLShaderProgram m_shader1
  OpenGL shading pipeline for points.
- QOpenGLShaderProgram m_shader2
  OpenGL shading pipeline for squares.
- QOpenGLShaderProgram m_shader3
  OpenGL shading pipeline for cubes.
- QOpenGLBuffer * m_verticesVBO
  vertices buffer (GPU)
- QVector< float > m_vertices
  vertices buffer (CPU)
- unsigned int m_maxPoints
  maximum number of points allowed (# vertices / 3)
- unsigned int m_maxRender
  maximum number of points rendered (less than or equal to max points)
- float m_pointSize
  the size of points
- QColor m_color
  color of points
- Mode m_mode
  rendering mode
- bool m_shader1OK
  shader1 ok to use
- bool m_shader2OK
  shader2 ok to use
- bool m_shader3OK
  shader3 ok to use

6.47.1 Detailed Description

A class to represent a point cloud.

6.47.2 Member Enumeration Documentation

6.47.2.1 enum PointCloud::Mode

The rendering mode for the cloud.

Enumerator

Points

Squares render points as OpenGL points

Cubes render points as squares

6.47.3 Constructor & Destructor Documentation

6.47.3.1 PointCloud::PointCloud (LangmuirViewer & *viewer*, QObject * *parent* = 0) [explicit]

create the PointCloud

Parameters

viewer	the viewer
parent	QObject this belongs to

6.47.3.2 PointCloud::~PointCloud ( )

destroy the PointCloud

6.47.4 Member Function Documentation

6.47.4.1 void PointCloud::colorChanged (QColor *color*) [signal]

signal that the color of has changed

Parameters

color	value of color

6.47.4.2 virtual void PointCloud::draw ( ) [protected], [virtual]

perform OpenGL drawing operations

Reimplemented from SceneObject.

6.47.4.3 void PointCloud::drawCubes ( ) [protected]

render function

6.47.4.4 void PointCloud::drawFallback ( ) [protected]

render function

6.47.4.5 void PointCloud::drawPoints ( ) [protected]

render function

6.47.4.6 void PointCloud::drawSquares ( ) [protected]

render function

6.47.4.7 const QColor& PointCloud::getColor ( ) const

get color

6.47.4.8 unsigned int PointCloud::getMaxPoints ( )

get the maximum number of points that can be rendered

6.47.4.9 unsigned int PointCloud::getMaxRender ( )

get the number of points currently being rendered

Warning

max points >= max rendered

6.47.4.10 Mode PointCloud::getMode ( ) const

get render mode

6.47.4.11 float PointCloud::getPointSize ( ) const

get x length

6.47.4.12 virtual void PointCloud::init ( ) [protected], [virtual]

initialize object

Reimplemented from SceneObject.

6.47.4.13 void PointCloud::initShaders ( ) [protected]

load the shaders

6.47.4.14 virtual void PointCloud::makeConnections ( ) [virtual], [slot]

make signal/slot connections

6.47.4.15 void PointCloud::maxPointsChanged (unsigned int *value*) [signal]

signal that the maximum number of points has changed

Parameters

value	value of max

6.47.4.16 void PointCloud::maxRenderChanged (unsigned int *value*) [signal]

signal that the maximum number of points rendered has changed

Parameters

value	value of max

6.47.4.17 void PointCloud::modeChanged (PointCloud::Mode *mode*) [signal]

signal that the render mode has changed

Parameters

mode	value of rendering mode

6.47.4.18 void PointCloud::modeChanged (QString *mode*) [signal]

signal that the render mode has changed

Parameters

mode	value of rendering mode

6.47.4.19 static QString PointCloud::modeToQString (Mode *mode*) [static], [slot]

convert Mode to string

Parameters

mode	mode enum

6.47.4.20 void PointCloud::pointSizeChanged (float *value*) [signal]

signal that the point size has changed

Parameters

value	value of point size

6.47.4.21 static Mode PointCloud::QStringToMode (QString *string*) [static], [slot]

convert string to Mode enum

Parameters

string	mode string

6.47.4.22 void PointCloud::setColor (QColor *color*) [slot]

set the color

Parameters

color	color to set

6.47.4.23 void PointCloud::setMaxPoints (unsigned int *value*) [slot]

change the maximum number of points allowed

Parameters

value	max points to set

This deletes the GPU memory and creates a new buffer. Do not do it often.

6.47.4.24 void PointCloud::setMaxRender (unsigned int *value*) [slot]

change the maximum number of points to render

Parameters

value	max points to render

Make sure this is less than the max points allowed.

6.47.4.25 void PointCloud::setMode (Mode *mode*) [slot]

set the mode

Parameters

mode	mode to set

6.47.4.26 void PointCloud::setPointSize (float *value*) [slot]

set the point size

Parameters

value	point size to set

6.47.4.27 void PointCloud::updateVBO ( ) [slot]

update the GPU memory using CPU vertices

First alter the CPU vertices, set the max render, then update the VBO.

6.47.4.28 QVector<float>& PointCloud::vertices ( )

get the list of vertices (CPU memory)

Warning

please do not resize this vector

6.47.5 Member Data Documentation

6.47.5.1 QColor PointCloud::m_color [protected]

color of points

6.47.5.2 unsigned int PointCloud::m_maxPoints [protected]

maximum number of points allowed (# vertices / 3)

6.47.5.3 unsigned int PointCloud::m_maxRender [protected]

maximum number of points rendered (less than or equal to max points)

6.47.5.4 Mode PointCloud::m_mode [protected]

rendering mode

6.47.5.5 float PointCloud::m_pointSize [protected]

the size of points

6.47.5.6 QOpenGLShaderProgram PointCloud::m_shader1 [protected]

OpenGL shading pipeline for points.

6.47.5.7 bool PointCloud::m_shader1OK [protected]

shader1 ok to use

6.47.5.8 QOpenGLShaderProgram PointCloud::m_shader2 [protected]

OpenGL shading pipeline for squares.

6.47.5.9 bool PointCloud::m_shader2OK [protected]

shader2 ok to use

6.47.5.10 QOpenGLShaderProgram PointCloud::m_shader3 [protected]

OpenGL shading pipeline for cubes.

6.47.5.11 bool PointCloud::m_shader3OK [protected]

shader3 ok to use

6.47.5.12 QVector<float> PointCloud::m_vertices [protected]

vertices buffer (CPU)

6.47.5.13 QOpenGLBuffer* PointCloud::m_verticesVBO [protected]

vertices buffer (GPU)

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

- /home/adam/opt/langmuir/src/langmuirView/include/pointcloud.h

6.48 PointDialog Class Reference

`#include <pointdialog.h>`

Public Slots

- void init ()
- void update ()
- void remember ()
- void reset ()
- void updateComboBoxElectrons (PointCloud::Mode mode)
- void updateComboBoxDefects (PointCloud::Mode mode)
- void updateComboBoxHoles (PointCloud::Mode mode)
- void updateComboBoxTraps (PointCloud::Mode mode)
- void updateSpinBoxElectrons (float d)
- void updateSpinBoxDefects (float d)
- void updateSpinBoxHoles (float d)
- void updateSpinBoxTraps (float d)
- void updateCheckBoxElectrons (bool checked)
- void updateCheckBoxDefects (bool checked)
- void updateCheckBoxHoles (bool checked)
- void updateCheckBoxTraps (bool checked)
- void on_comboBoxElectrons_currentTextChanged (const QString &text)
- void on_comboBoxDefects_currentTextChanged (const QString &text)
- void on_comboBoxHoles_currentTextChanged (const QString &text)
- void on_comboBoxTraps_currentTextChanged (const QString &text)
- void on_spinBoxElectrons_valueChanged (double d)
- void on_spinBoxDefects_valueChanged (double d)
- void on_spinBoxHoles_valueChanged (double d)
- void on_spinBoxTraps_valueChanged (double d)
- void on_checkBoxElectrons_stateChanged (int state)
- void on_checkBoxDefects_stateChanged (int state)
- void on_checkBoxHoles_stateChanged (int state)
- void on_checkBoxTraps_stateChanged (int state)
- void on_pushButtonReset_clicked ()
- void on_buttonBox_rejected ()

Public Member Functions

- PointDialog (LangmuirViewer &viewer, QWidget *parent=0)
- ~PointDialog ()

Private Attributes

- Ui::PointDialog * ui
- LangmuirViewer & m_viewer
- float e_pointSize_old
- float d_pointSize_old
- float h_pointSize_old
- float t_pointSize_old
- PointCloud::Mode e_mode_old
- PointCloud::Mode d_mode_old
- PointCloud::Mode h_mode_old
- PointCloud::Mode t_mode_old
- bool e_visible
- bool d_visible
- bool h_visible
- bool t_visible

6.48.1 Constructor & Destructor Documentation

6.48.1.1 PointDialog::PointDialog (LangmuirViewer & *viewer,* QWidget * *parent* = 0) [explicit]

6.48.1.2 PointDialog::~PointDialog ( )

6.48.2 Member Function Documentation

6.48.2.1 void PointDialog::init ( ) [slot]

6.48.2.2 void PointDialog::on_buttonBox_rejected ( ) [slot]

6.48.2.3 void PointDialog::on_checkBoxDefects_stateChanged (int *state*) [slot]

6.48.2.4 void PointDialog::on_checkBoxElectrons_stateChanged (int *state*) [slot]

6.48.2.5 void PointDialog::on_checkBoxHoles_stateChanged (int *state*) [slot]

6.48.2.6 void PointDialog::on_checkBoxTraps_stateChanged (int *state*) [slot]

6.48.2.7 void PointDialog::on_comboBoxDefects_currentTextChanged (const QString & *text*) [slot]

6.48.2.8 void PointDialog::on_comboBoxElectrons_currentTextChanged (const QString & *text*) [slot]

6.48.2.9 void PointDialog::on_comboBoxHoles_currentTextChanged (const QString & *text*) [slot]

6.48.2.10 void PointDialog::on_comboBoxTraps_currentTextChanged (const QString & *text*) [slot]

6.48.2.11 void PointDialog::on_pushButtonReset_clicked ( ) [slot]

6.48.2.12 void PointDialog::on_spinBoxDefects_valueChanged (double *d*) [slot]

6.48.2.13 void PointDialog::on_spinBoxElectrons_valueChanged (double *d*) [slot]

6.48.2.14 void PointDialog::on_spinBoxHoles_valueChanged (double *d*) [slot]

6.48.2.15 void PointDialog::on_spinBoxTraps_valueChanged (double *d*) [slot]

6.48.2.16 void PointDialog::remember ( ) [slot]

6.48.2.17 void PointDialog::reset ( ) [slot]

6.48.2.18 void PointDialog::update ( ) [slot]

6.48.2.19 void PointDialog::updateCheckBoxDefects (bool *checked*) [slot]

6.48.2.20 void PointDialog::updateCheckBoxElectrons (bool *checked*) [slot]

6.48.2.21 void PointDialog::updateCheckBoxHoles (bool *checked*) [slot]

6.48.2.22 void PointDialog::updateCheckBoxTraps (bool *checked*) [slot]

6.48.2.23 void PointDialog::updateComboBoxDefects (**PointCloud::Mode** *mode*) [slot]

6.48.2.24 void PointDialog::updateComboBoxElectrons (**PointCloud::Mode** *mode*) [slot]

6.48.2.25 void PointDialog::updateComboBoxHoles (**PointCloud::Mode** *mode*) [slot]

6.48.2.26 void PointDialog::updateComboBoxTraps (**PointCloud::Mode** *mode*) [slot]

6.48.2.27 void PointDialog::updateSpinBoxDefects (float *d*) [slot]

6.48.2.28 void PointDialog::updateSpinBoxElectrons (float *d*) [slot]

6.48.2.29 void PointDialog::updateSpinBoxHoles (float *d*) [slot]

6.48.2.30 void PointDialog::updateSpinBoxTraps (float *d*) [slot]

6.48.3 Member Data Documentation

6.48.3.1 **PointCloud::Mode** PointDialog::d_mode_old [private]

6.48.3.2 float PointDialog::d_pointSize_old [private]

6.48.3.3 bool PointDialog::d_visible [private]

6.48.3.4 **PointCloud::Mode** PointDialog::e_mode_old [private]

6.48.3.5 float PointDialog::e_pointSize_old [private]

6.48.3.6 bool PointDialog::e_visible [private]

6.48.3.7 **PointCloud::Mode** PointDialog::h_mode_old [private]

6.48.3.8 float PointDialog::h_pointSize_old [private]

6.48.3.9 bool PointDialog::h_visible [private]

6.48.3.10 **LangmuirViewer**& PointDialog::m_viewer [private]

6.48.3.11 **PointCloud::Mode** PointDialog::t_mode_old [private]

6.48.3.12 float PointDialog::t_pointSize_old [private]

6.48.3.13 bool PointDialog::t_visible [private]

6.48.3.14 Ui::PointDialog* PointDialog::ui [private]

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

- /home/adam/opt/langmuir/src/langmuirView/include/pointdialog.h

6.49 Langmuir::Potential Class Reference

A class to calculate the potential.

`#include <potential.h>`

Public Member Functions

- Potential (World &world, QObject *parent=0)
  Potential Create the potential.
- void setPotentialZero ()
  sets the value of the potential to zero at every grid site
- void setPotentialLinear ()
  Adds a linear potential calcualted from voltage.left and voltage.right along the x-direction.
- void setPotentialGate ()
  Adds a linear potential calculated from slope.z along the z-direction.
- void setPotentialTraps (const QList< int > &trapIDs=QList< int >(), const QList< double > &trapPotentials=QList< double >())
  Adds shifts to the potential at the various sites.
- void precalculateArrays ()
  pre-calculates r2, r, and 1/r
- void updateCouplingConstants ()
  pre-calculates coupling constants
- double coulombE (int site_i)
  calculates Coulomb potential from electrons at specific grid site
- double coulombImageE (int site_i)
  calculates Coulomb image-potential from electrons at specific grid site
- double gaussE (int site_i)
  calculates Coulomb potential from electrons at specific grid site, assuming gaussians
- double gaussImageE (int site_i)
  calculates Coulomb image-potential from electrons at specific grid site, assuming gaussians
- double coulombH (int site_i)
  calculates Coulomb potential from holes at specific grid site
- double coulombImageH (int site_i)
  calculates Coulomb image-potential from holes at specific grid site
- double gaussH (int site)
  calculates Coulomb potential from holes at specific grid site, assuming gaussians
- double gaussImageH (int site)
  calculates Coulomb image-potential from holes at specific grid site, assuming gaussians
- double coulombD (int site_i)
  calculates Coulomb potential from charged defects at specific grid site
- double coulombImageD (int site_i)
  calculates Coulomb image-potential from charged defects at specific grid site

- double gaussD (int site_i)

  calculates Coulomb potential from charged defects at specific grid site, assuming gaussians
- double gaussImageD (int site_i)

  calculates Coulomb image-potential from charged defects at specific grid site, assuming gaussians

Private Attributes

- World & m_world

  reference to the World

6.49.1 Detailed Description

A class to calculate the potential.

6.49.2 Constructor & Destructor Documentation

6.49.2.1 Langmuir::Potential::Potential (World & *world,* QObject * *parent* = 0)

Potential Create the potential.

Parameters

world	reference to the World
parent	QObject this belongs to

6.49.3 Member Function Documentation

6.49.3.1 double Langmuir::Potential::coulombD (int *site_i*)

calculates Coulomb potential from charged defects at specific grid site

Parameters

site	the site of interest

6.49.3.2 double Langmuir::Potential::coulombE (int *site_i*)

calculates Coulomb potential from electrons at specific grid site

Parameters

site	the site of interest

6.49.3.3 double Langmuir::Potential::coulombH (int *site_i*)

calculates Coulomb potential from holes at specific grid site

Parameters

site	the site of interest

6.49.3.4 double Langmuir::Potential::coulombImageD (int *site_i*)

calculates Coulomb image-potential from charged defects at specific grid site

Parameters

site	the site of interest

6.49.3.5 double Langmuir::Potential::coulombImageE (int *site_i*)

calculates Coulomb image-potential from electrons at specific grid site

Parameters

site	the site of interest

6.49.3.6 double Langmuir::Potential::coulombImageH (int *site_i*)

calculates Coulomb image-potential from holes at specific grid site

Parameters

site	the site of interest

6.49.3.7 double Langmuir::Potential::gaussD (int *site_i*)

calculates Coulomb potential from charged defects at specific grid site, assuming gaussians

Parameters

site	the site of interest

6.49.3.8 double Langmuir::Potential::gaussE (int *site_i*)

calculates Coulomb potential from electrons at specific grid site, assuming gaussians

Parameters

site	the site of interest

6.49.3.9 double Langmuir::Potential::gaussH (int *site*)

calculates Coulomb potential from holes at specific grid site, assuming gaussians

Parameters

site	the site of interest

6.49.3.10 double Langmuir::Potential::gaussImageD (int *site_i*)

calculates Coulomb image-potential from charged defects at specific grid site, assuming gaussians

Parameters

site	the site of interest

6.49.3.11 double Langmuir::Potential::gaussImageE (int *site_i*)

calculates Coulomb image-potential from electrons at specific grid site, assuming gaussians

Parameters

site	the site of interest

6.49.3.12 double Langmuir::Potential::gaussImageH (int *site*)

calculates Coulomb image-potential from holes at specific grid site, assuming gaussians

Parameters

site	the site of interest

6.49.3.13 void Langmuir::Potential::precalculateArrays ( )

pre-calculates r2, r, and 1/r

6.49.3.14 void Langmuir::Potential::setPotentialGate ( )

Adds a linear potential calculated from slope.z along the z-direction.

6.49.3.15 void Langmuir::Potential::setPotentialLinear ( )

Adds a linear potential calcualted from voltage.left and voltage.right along the x-direction.

6.49.3.16 void Langmuir::Potential::setPotentialTraps (const QList< int > & *trapIDs* = QList< int >(), const QList< double > & *trapPotentials* = QList< double >())

Adds shifts to the potential at the various sites.

Parameters

trapIDs	list of site ids
trapPotentials	list of shifts

6.49.3.17 void Langmuir::Potential::setPotentialZero ( )

sets the value of the potential to zero at every grid site

6.49.3.18 void Langmuir::Potential::updateCouplingConstants ( )

pre-calculates coupling constants

6.49.4 Member Data Documentation

6.49.4.1 World& Langmuir::Potential::m_world [private]

reference to the World

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

- /home/adam/opt/langmuir/src/langmuirCore/include/potential.h

6.50 Langmuir::Random Class Reference

A class to generate random numbers.

`#include <rand.h>`

Public Member Functions

- Random (quint64 seed=0, QObject *parent=0)
  Random.
- ~Random ()
  Destroy objects.
- quint64 seed ()
  Get the seed that was used.
- void seed (quint64 seed)
  seed the generator (again)
- double random ()
  Generate a random double from the uniform distribution [0, 1].
- double range (const double low=0.0, const double high=1.0)
  Generate a random double from the uniform distribution [low, high].
- double normal (const double mean, const double sigma)
  Generate a random double from the normal distribution.
- int integer (const int low=0, const int high=1)
  Generate a random int from the uniform distribution [low, high].
- bool metropolis (double energyChange, double inversekT)
  Randomly choose yes using a Boltzmann factor.
- bool metropolisWithCoupling (double energyChange, double inversekT, double coupling)
  Randomly choose yes using a Boltzmann factor and coupling constant.
- bool chooseYes (double percent)
  Randomly choose yes a percent of the time.
- bool chooseNo (double percent)
  Randomly choose no a percent of the time.

Private Attributes

- boost::mt19937 * twister
  The underlying random number generator.
- boost::variate_generator
  < boost::mt19937
  &, boost::uniform_01< double > > * generator01
  The underlying generator coupled to the uniform distribution on [0,1].
- quint64 m_seed
  The seed used to start the generator.

Friends

- QDataStream & operator<< (QDataStream &stream, Random &random)
  Output the random state to a QDataStream ***Possibly Broken****.*
- QDataStream & operator>> (QDataStream &stream, Random &random)
  Load the random state from a QDataStream ***Possibly Broken****.*
- QTextStream & operator<< (QTextStream &stream, Random &random)

Output the random state to a QTextStream ***Possibly Broken****.*

- QTextStream & operator>> (QTextStream &stream, Random &random)

  Load the random state from a QTextStream ***Possibly Broken****.*

- std::ostream & operator<< (std::ostream &stream, Random &random)

  Output the random state to a std::ostream.

- std::istream & operator>> (std::istream &stream, Random &random)

  Load the random state from a std::istream.

6.50.1 Detailed Description

A class to generate random numbers.

6.50.2 Constructor & Destructor Documentation

6.50.2.1 Langmuir::Random::Random (quint64 *seed* = 0, QObject * *parent* = 0)

Random.

Parameters

seed	makes the generator deterministic • seed == 0 uses the current clock time
parent	object this belongs to

6.50.2.2 Langmuir::Random::~Random ( )

Destroy objects.

6.50.3 Member Function Documentation

6.50.3.1 bool Langmuir::Random::chooseNo (double *percent*)

Randomly choose no a percent of the time.

6.50.3.2 bool Langmuir::Random::chooseYes (double *percent*)

Randomly choose yes a percent of the time.

6.50.3.3 int Langmuir::Random::integer (const int *low* = 0, const int *high* = 1)

Generate a random int from the uniform distribution [low, high].

6.50.3.4 bool Langmuir::Random::metropolis (double *energyChange*, double *inversekT*)

Randomly choose yes using a Boltzmann factor.

Parameters

energyChange	change in energy when going from initial to final state
inversekT	decay constant in exponential

6.50.3.5 bool Langmuir::Random::metropolisWithCoupling (double *energyChange,* double *inversekT,* double *coupling*)

Randomly choose yes using a Boltzmann factor and coupling constant.

Parameters

energyChange	change in energy when going from initial to final state
inversekT	decay constant in exponential
coupling	alters acceptance probability • if energy $<$ 0 : chooses yes coupling $*$ Boltzmann factor percent of the time • if energy $>$ 0 : chooses yes 1 - coupling percent of the time

6.50.3.6 double Langmuir::Random::normal (const double *mean,* const double *sigma*)

Generate a random double from the normal distribution.

Parameters

mean	average value of the normal distribution sampled
sigma	standard deviation of the normal distribution sampled

6.50.3.7 double Langmuir::Random::random ( )

Generate a random double from the uniform distribution [0, 1].

6.50.3.8 double Langmuir::Random::range (const double *low* = `0.0`, const double *high* = `1.0`)

Generate a random double from the uniform distribution [low, high].

6.50.3.9 quint64 Langmuir::Random::seed ( )

Get the seed that was used.

6.50.3.10 void Langmuir::Random::seed (quint64 *seed*)

seed the generator (again)

If seed == 0 is used, then the generator **does not** use the current time. This is different than the constructor Random.

6.50.4 Friends And Related Function Documentation

6.50.4.1 QDataStream& operator<< (QDataStream & *stream,* Random & *random*) [friend]

Output the random state to a QDataStream **Possibly Broken**.

Warning

This may not quite be working correctly

6.50.4.2 QTextStream& operator<< (QTextStream & *stream,* Random & *random*) [friend]

Output the random state to a QTextStream **Possibly Broken**.

Warning

This may not quite be working correctly

6.50.4.3 std::ostream& operator<< (std::ostream & *stream,* Random & *random*) [friend]

Output the random state to a std::ostream.

6.50.4.4 QDataStream& operator>> (QDataStream & *stream,* Random & *random*) [friend]

Load the random state from a QDataStream **Possibly Broken**.

Warning

This may not quite be working correctly

6.50.4.5 QTextStream& operator>> (QTextStream & *stream,* Random & *random*) [friend]

Load the random state from a QTextStream **Possibly Broken**.

Warning

This may not quite be working correctly

6.50.4.6 std::istream& operator>> (std::istream & *stream,* Random & *random*) [friend]

Load the random state from a std::istream.

6.50.5 Member Data Documentation

6.50.5.1 boost::variate_generator<boost::mt19937&, boost::uniform_01<double> >* Langmuir::Random::generator01 [private]

The underlying generator coupled to the uniform distribution on [0,1].

6.50.5.2 quint64 Langmuir::Random::m_seed [private]

The seed used to start the generator.

6.50.5.3 boost::mt19937* Langmuir::Random::twister [private]

The underlying random number generator.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/rand.h

6.51 Langmuir::RecombinationAgent Class Reference

A class to remove Excitons.

`#include <drainagent.h>`

Public Member Functions

- RecombinationAgent (World &world, QObject *parent=0)

  create a RecombinationAgent
- virtual bool tryToAccept (ChargeAgent *charge)

  accept charge with constant probability
- void guessProbability ()

  calculate an acceptance probability based upon the desired rate and encounter frequency

Private Member Functions

- virtual double energyChange (int fSite)

  currently implemented as zero and not really used

Additional Inherited Members

6.51.1 Detailed Description

A class to remove Excitons.

Currently, the RecombinationAgent is not really doing much of anything outside of keeping track of recombination statistics. The meat of the recombination resides in Simulation::performRecombinations(). This should probably be changes in the future.

6.51.2 Constructor & Destructor Documentation

6.51.2.1 Langmuir::RecombinationAgent::RecombinationAgent (World & *world,* QObject * *parent* = `0`)

create a RecombinationAgent

6.51.3 Member Function Documentation

6.51.3.1 virtual double Langmuir::RecombinationAgent::energyChange (int *fSite*) `[private],[virtual]`

currently implemented as zero and not really used

Reimplemented from Langmuir::FluxAgent.

6.51.3.2 void Langmuir::RecombinationAgent::guessProbability ( )

calculate an acceptance probability based upon the desired rate and encounter frequency

6.51.3.3 virtual bool Langmuir::RecombinationAgent::tryToAccept (ChargeAgent * *charge*) [virtual]

accept charge with constant probability

Reimplemented from Langmuir::DrainAgent.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/drainagent.h

6.52 Langmuir::RecordDialog Class Reference

`#include <gridview.h>`

Public Slots

- void openFileDialog ()
- void setWork (QString)
- void setStub (QString)
- void setType (QString)
- void setCount (int)
- void setQuality (int)

Signals

- void workChanged (QString value)
- void stubChanged (QString value)
- void typeChanged (QString value)
- void countChanged (int value)
- void qualityChanged (int value)

Public Member Functions

- RecordDialog (QWidget *parent=0)

Public Attributes

- QGridLayout * gridLayout
- QLabel * L1
- QLabel * L2
- QLabel * L4
- QLabel * L5
- QLabel * L6
- QLineEdit * LE1
- SSpinBox * SB2
- SSpinBox * SB3
- QComboBox * CB1
- QPushButton * PB1
- QDialogButtonBox * OK
- QDir work
- QString stub
- QString type
- int count
- int quality

6.52.1 Constructor & Destructor Documentation

6.52.1.1 Langmuir::RecordDialog::RecordDialog (QWidget * *parent* = 0)

6.52.2 Member Function Documentation

6.52.2.1 void Langmuir::RecordDialog::countChanged (int *value*) [signal]

6.52.2.2 void Langmuir::RecordDialog::openFileDialog ( ) [slot]

6.52.2.3 void Langmuir::RecordDialog::qualityChanged (int *value*) [signal]

6.52.2.4 void Langmuir::RecordDialog::setCount (int) [slot]

6.52.2.5 void Langmuir::RecordDialog::setQuality (int) [slot]

6.52.2.6 void Langmuir::RecordDialog::setStub (QString) [slot]

6.52.2.7 void Langmuir::RecordDialog::setType (QString) [slot]

6.52.2.8 void Langmuir::RecordDialog::setWork (QString) [slot]

6.52.2.9 void Langmuir::RecordDialog::stubChanged (QString *value*) [signal]

6.52.2.10 void Langmuir::RecordDialog::typeChanged (QString *value*) [signal]

6.52.2.11 void Langmuir::RecordDialog::workChanged (QString *value*) [signal]

6.52.3 Member Data Documentation

6.52.3.1 QComboBox* Langmuir::RecordDialog::CB1

6.52.3.2 int Langmuir::RecordDialog::count

6.52.3.3 QGridLayout* Langmuir::RecordDialog::gridLayout

6.52.3.4 QLabel* Langmuir::RecordDialog::L1

6.52.3.5 QLabel* Langmuir::RecordDialog::L2

6.52.3.6 QLabel* Langmuir::RecordDialog::L4

6.52.3.7 QLabel* Langmuir::RecordDialog::L5

6.52.3.8 QLabel* Langmuir::RecordDialog::L6

6.52.3.9 QLineEdit* Langmuir::RecordDialog::LE1

6.52.3.10 QDialogButtonBox* Langmuir::RecordDialog::OK

6.52.3.11 QPushButton* Langmuir::RecordDialog::PB1

6.52.3.12 int Langmuir::RecordDialog::quality

6.52.3.13 SSpinBox* Langmuir::RecordDialog::SB2

6.52.3.14 SSpinBox* Langmuir::RecordDialog::SB3

6.52.3.15 QString Langmuir::RecordDialog::stub

6.52.3.16 QString Langmuir::RecordDialog::type

6.52.3.17 QDir Langmuir::RecordDialog::work

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.53 SceneObject Class Reference

Base class for objects in OpenGL scene.

`#include <sceneobject.h>`

Public Slots

- void toggleVisible ()
  toggle visibility
- void setVisible (bool draw=true)
  set the visibility
- virtual void makeConnections ()
  make signal/slot connections

Signals

- void visibleChanged (bool drawn)
  signal that the visibility has changed

Public Member Functions

- SceneObject (LangmuirViewer &viewer, QObject *parent=0)
  create the SceneObject
- bool isVisible ()
  true if object is drawn
- void render ()
  calls OpenGL drawing commands.

Protected Member Functions

- virtual void init ()
  initialize object
- virtual void draw ()
  perform OpenGL drawing operations
- virtual void preDraw ()
  perform OpenGL drawing operations before draw()
- virtual void postDraw ()
  perform OpenGL drawing operations after draw()

Protected Attributes

- LangmuirViewer & m_viewer

  reference to OpenGL widget
- bool visible_

  visibility

6.53.1 Detailed Description

Base class for objects in OpenGL scene.

6.53.2 Constructor & Destructor Documentation

6.53.2.1 SceneObject::SceneObject (LangmuirViewer & *viewer*, QObject * *parent* = 0) [explicit]

create the SceneObject

Parameters

viewer	the viewer
parent	QObject this belongs to

6.53.3 Member Function Documentation

6.53.3.1 virtual void SceneObject::draw ( ) [protected], [virtual]

perform OpenGL drawing operations

Reimplemented in PointCloud, Light, Box, Mesh, Axis, and Grid.

6.53.3.2 virtual void SceneObject::init ( ) [protected], [virtual]

initialize object

Explicitly call this in derived class constructor.

Reimplemented in PointCloud, Light, Box, Mesh, Axis, CornerAxis, and Grid.

6.53.3.3 bool SceneObject::isVisible ( )

true if object is drawn

6.53.3.4 virtual void SceneObject::makeConnections ( ) [virtual], [slot]

make signal/slot connections

6.53.3.5 virtual void SceneObject::postDraw ( ) [protected], [virtual]

perform OpenGL drawing operations after draw()

Reimplemented in CornerAxis.

6.53.3.6 virtual void SceneObject::preDraw ( ) [protected], [virtual]

perform OpenGL drawing operations before draw()

Reimplemented in CornerAxis.

6.53.3.7 void SceneObject::render ( )

calls OpenGL drawing commands.

Use this inside the paintGL() or draw() functions of the main OpenGL widget.

6.53.3.8 void SceneObject::setVisible (bool *draw* = true) [slot]

set the visibility

Parameters

draw	true if object is to be drawn

6.53.3.9 void SceneObject::toggleVisible ( ) [slot]

toggle visibility

6.53.3.10 void SceneObject::visibleChanged (bool *drawn*) [signal]

signal that the visibility has changed

Parameters

drawn	true if object is visible

6.53.4 Member Data Documentation

6.53.4.1 LangmuirViewer& SceneObject::m_viewer [protected]

reference to OpenGL widget

6.53.4.2 bool SceneObject::visible_ [protected]

visibility

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

- /home/adam/opt/langmuir/src/langmuirView/include/sceneobject.h

6.54 Langmuir::SceneOptions Class Reference

`#include <gridview.h>`

Public Member Functions

- SceneOptions (QWidget *parent)

Public Attributes

- QGridLayout * layout
- QColorDialog * colorDialog
- QList< Button * > buttons
- QList< QLabel * > labels
- QList< DSpinBox * > spinBoxes
- QList< CheckBox * > checkBoxes

6.54.1 Constructor & Destructor Documentation

6.54.1.1 Langmuir::SceneOptions::SceneOptions (QWidget * *parent*)

6.54.2 Member Data Documentation

6.54.2.1 QList< Button* > Langmuir::SceneOptions::buttons

6.54.2.2 QList< CheckBox* > Langmuir::SceneOptions::checkBoxes

6.54.2.3 QColorDialog* Langmuir::SceneOptions::colorDialog

6.54.2.4 QList< QLabel* > Langmuir::SceneOptions::labels

6.54.2.5 QGridLayout* Langmuir::SceneOptions::layout

6.54.2.6 QList< DSpinBox* > Langmuir::SceneOptions::spinBoxes

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.55 Langmuir::Simulation Class Reference

A class to orchestrate the calculation.

`#include <simulation.h>`

Public Member Functions

- Simulation (World &world, QObject *parent=0)

  Create a Simulation.
- virtual ~Simulation ()

  Destroy the Simulation.
- virtual void performIterations (int nIterations)

  simulate for a set number of steps

Protected Member Functions

- void performRecombinations ()

  Recombine holes and electrons (in solarcell simulations only)
- void performInjections ()

  Tell sources to inject charges.

- void balanceCharges ()

  Try *to use the sources to keep the number of ChargeAgents balanced*

- void nextTick ()

  Remove charges from the simulation.

Static Protected Member Functions

- static void chargeAgentCoulombInteractionQtConcurrentCPU (ChargeAgent *chargeAgent)

  A method needed to call ChargeAgent::coulombCPU() in parallel.

- static void chargeAgentCoulombInteractionQtConcurrentGPU (ChargeAgent *chargeAgent)

  A method needed to call ChargeAgent::coulombGPU() in parallel.

Protected Attributes

- World & m_world

  Reference to World object.

6.55.1 Detailed Description

A class to orchestrate the calculation.

6.55.2 Constructor & Destructor Documentation

6.55.2.1 Langmuir::Simulation::Simulation (World & *world*, QObject * *parent* = 0)

Create a Simulation.

Parameters

world	reference to World Object
parent	QObject this belongs to

6.55.2.2 virtual Langmuir::Simulation::~Simulation ( ) [virtual]

Destroy the Simulation.

6.55.3 Member Function Documentation

6.55.3.1 void Langmuir::Simulation::balanceCharges ( ) [protected]

Try to use the sources to keep the number of ChargeAgents balanced

6.55.3.2 static void Langmuir::Simulation::chargeAgentCoulombInteractionQtConcurrentCPU (ChargeAgent * *chargeAgent*) [static], [protected]

A method needed to call ChargeAgent::coulombCPU() in parallel.

6.55.3.3 static void Langmuir::Simulation::chargeAgentCoulombInteractionQtConcurrentGPU (ChargeAgent * *chargeAgent*) [static], [protected]

A method needed to call ChargeAgent::coulombGPU() in parallel.

Does not perform GPU calcuations. The coulomb kernel in OpenCLHelper is used to do that. This function copies the GPU results from OpenCLHelper to each ChargeAgent. It is assumed that the coulomb kernel was launched beforehand.

6.55.3.4 void Langmuir::Simulation::nextTick ( ) [protected]

Remove charges from the simulation.

Charges are removed only if a DrainAgent sets their removed status to True. This function will also output carrier statistics if output.id.on.delete is set.

6.55.3.5 void Langmuir::Simulation::performInjections ( ) [protected]

Tell sources to inject charges.

6.55.3.6 virtual void Langmuir::Simulation::performIterations (int *nIterations*) [virtual]

simulate for a set number of steps

Parameters

nIterations	the number of steps to simulate

6.55.3.7 void Langmuir::Simulation::performRecombinations ( ) [protected]

Recombine holes and electrons (in solarcell simulations only)

6.55.4 Member Data Documentation

6.55.4.1 World& Langmuir::Simulation::m_world [protected]

Reference to World object.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/simulation.h

6.56 Langmuir::SimulationParameters Struct Reference

A struct to store all simulation options To add new variables, follow these steps:

`#include <parameters.h>`

Public Member Functions

- SimulationParameters ()

Public Attributes

- QString simulationType

  tells Langmuir how to set up the Sources and Drains: ("transistor", "solarcell")
- quint64 randomSeed

  seed the random number generator, if negative, uses the current time (making seperate runs random)
- qint32 gridZ

  the number of sites per layer, at least one
- qint32 gridY

  the number of sites along the device width, at least one
- qint32 gridX

  the number of sites along the device length, at least one
- bool coulombCarriers

  turn on Coulomb interactions between ChargeAgents
- qreal coulombGaussianSigma

  multiply Coulomb terms by erf[r/(sigma sqrt[2])]; nothing happens if its zero
- qint32 defectsCharge

  the charge of defect sites
- qint32 outputXyz

  output trajectory file (if $n < 0$, only at the end; if n == 0, never; if $n > 0$, every $n *$ iterations.print steps)
- bool outputXyzE

  output electrons in trajectory file (ignored if outputXyz is off)
- bool outputXyzH

  output holes in trajectory file (ignored if outputXyz is off)
- bool outputXyzD

  output defects in trajectory file (ignored if outputXyz is off)
- bool outputXyzT

  output traps in trajectory file (ignored if outputXyz is off)
- qint32 outputXyzMode

  output mode for xyz file (if 0, particle count varies; if 1, particle count is constant using "phantom particles")
- bool outputIdsOnDelete

  output carrier lifetime and pathlength when they are deleted
- qint32 outputCoulomb

  output coulomb energy for the entire grid (if $n < 0$, only at the end; if n == 0, never; if $n > 0$, every $n *$ iterations.print steps)
- qint32 outputStepChk

  output a checkpoint file every this $*$ iterationsPrint
- bool outputChkTrapPotential

  output trap potentials in checkpoint files
- bool outputPotential

  output grid potential at the start of the simulation, includes the trap potential
- bool outputIsOn

  if false, produce no output (useful for LangmuirView)
- bool imageDefects

  output images of defects
- bool imageTraps

  output images of defects
- qint32 imageCarriers

  output images of carriers (if $n < 0$, only at the end; if n == 0, never; if $n > 0$, every $n *$ iterations.print steps)
- qint32 iterationsPrint

  if Langmuir, how often to output; if LangmuirView, how many steps between rendering

- qint32 iterationsReal

  number of simulation steps after equilibration
- qint32 outputPrecision

  number of significant figures used for doubles in output
- qint32 outputWidth

  width of columns in output, ignored in certain files, like trajectory files
- QString outputStub

  the stub to use when naming output files
- qreal electronPercentage

  the percent of the grid that is reserved for electrons, between 0 and 1
- qreal holePercentage

  the percent of the grid that is reserved for holes, between 0 and 1
- qreal seedCharges

  if true, place charges randomly before the simulation starts
- qreal defectPercentage

  the percent of the grid that is reserved for electrons, between 0 and 1
- qreal trapPercentage

  the percent of the grid that is reserved for traps, between 0 and 1
- qreal trapPotential

  the potential of traps
- qreal gaussianStdev

  the standard deviation of trap sites
- qreal seedPercentage

  the percent of the traps to be placed and grown upon to form islands
- qreal voltageRight

  the potential on the right side of the grid, used in setting up an electric field
- qreal voltageLeft

  the potential on the left side of the grid, used in setting up an electric field
- qreal excitonBinding

  the energy change (eV) when a hole/electron pair goes from the same site to adjacent sites
- qreal temperatureKelvin

  the temperature used in the boltzmann factor
- qreal sourceRate

  the rate at which all sources inject charges
- qreal eSourceLRate

  the rate at which the left electron source injects charges (overrides default)
- qreal eSourceRRate

  the rate at which the right electron source injects charges (overrides default)
- qreal hSourceLRate

  the rate at which the left hole source injects charges (overrides default)
- qreal hSourceRRate

  the rate at which the right hole source injects charges (overrides default)
- qreal generationRate

  the rate at which the exciton source injects charges (overrides default)
- bool balanceCharges

  if true, try to keep the number of charges in the simulation balanced
- qreal drainRate

  the rate at which all drains accept charges (default, used when eDrainL, etc. are < 0)
- qreal eDrainLRate

  the rate at which the left electron drain accepts charges (overrides default)
- qreal eDrainRRate

the rate at which the right electron drain accepts charges (overrides default)

- qreal hDrainLRate

  the rate at which the left hole drain accepts charges (overrides default)
- qreal hDrainRRate

  the rate at which the right hole drain accepts charges (overrides default)
- bool useOpenCL

  if true, try to use OpenCL to speed up Coulomb interaction calculations
- qint32 workX

  the x size of OpenCL 3DRange kernel work groups - only needed if using SimulationParameters::outputCoulomb
- qint32 workY

  the y size of OpenCL 3DRange kernel work groups - only needed if using SimulationParameters::outputCoulomb
- qint32 workZ

  the z size of OpenCL 3DRange kernel work groups - only needed if using SimulationParameters::outputCoulomb
- qint32 workSize

  the size of OpenCL 1DRange kernel work groups
- qint32 openclThreshold

  the minimum number of charges that must be present to use OpenCL
- qint32 openclDeviceID

  the device to choose if there are multiple
- qreal boltzmannConstant

  physical constant, the boltzmann constant
- qreal dielectricConstant

  physical constant, the dielectic constant
- qreal elementaryCharge

  physical constant, the elementary charge
- qreal permittivitySpace

  physical constant, the permittivity of free spece
- qreal gridFactor

  size constant, the size associated with grid sites (∼1nm)
- qint32 electrostaticCutoff

  the cut off for Coulomb interations
- qreal electrostaticPrefactor

  a compilation of physical constants
- qreal inverseKT

  a compilation of physical constants
- bool okCL

  if true, OpenCL can be used on this platform
- quint32 currentStep

  the current step of the simulation
- QDateTime simulationStart

  the time this simulation started
- qint32 hoppingRange

  the number of sites away from a given site used when calculating neighboring sites
- qreal slopeZ

  slope of potential along z direction when there are multiple layers (as if there were a gate electrode)
- bool sourceMetropolis

  if true, use an energy change (source potential and site) and metropolis criterion to calculate injection probability for hole and electron sources (not exciton sources)
- bool sourceCoulomb

  if true, use the Coulomb + Image interaction when calcualting energy change for injection
- qreal recombinationRate

the rate at which holes and electrons can combine when they sit upon one another

- qint32 recombinationRange

  the number of sites to consider when performing recombinations (0 means same-site, 1 means one-site away and same-site)

- bool outputIdsOnEncounter

  output carrier lifetime and pathlength when holes and electrons encounter one another

- qreal sourceScaleArea

  for SimulationParameters::simulationType == "solarcell", multiply SimulationParameters::sourceRate by (Grid::xyPlaneArea)/(SimulationParameters::sourceScaleArea); if <= 0, does not scale rate

- qint32 maxThreads

  max threads allowed for QThreadPool - if its <= 0 then the QThread::idealThreadCount is used; note that Qt ignores PBS and SGE so when this isn't set Qt will use all the cores on a node

6.56.1 Detailed Description

A struct to store all simulation options To add new variables, follow these steps:

- declare the new variable in the SimulationParameters struct (parameters.h)
- assign the default value of the new variable in the SimulationParameters constructor (parameters.h)
- implement validity checking for the variable in the checkSimulationParameters() function (parameters.h)
- register the variable in the KeyValueParser constructor using the registerVariable() function (keyvalueparser.h)
- to use non-standard types, you must overload certain template functions in variable.h. See, for example, overloads for QDateTime in variable.h.

6.56.2 Constructor & Destructor Documentation

6.56.2.1 Langmuir::SimulationParameters::SimulationParameters ( ) [inline]

6.56.3 Member Data Documentation

6.56.3.1 bool Langmuir::SimulationParameters::balanceCharges

if true, try to keep the number of charges in the simulation balanced

6.56.3.2 qreal Langmuir::SimulationParameters::boltzmannConstant

physical constant, the boltzmann constant

6.56.3.3 bool Langmuir::SimulationParameters::coulombCarriers

turn on Coulomb interactions between ChargeAgents

6.56.3.4 qreal Langmuir::SimulationParameters::coulombGaussianSigma

multiply Coulomb terms by erf[r/(sigma sqrt[2])]; nothing happens if its zero

6.56.3.5 quint32 Langmuir::SimulationParameters::currentStep

the current step of the simulation

6.56.3.6 qreal Langmuir::SimulationParameters::defectPercentage

the percent of the grid that is reserved for electrons, between 0 and 1

6.56.3.7 qint32 Langmuir::SimulationParameters::defectsCharge

the charge of defect sites

6.56.3.8 qreal Langmuir::SimulationParameters::dielectricConstant

physical constant, the dielectic constant

6.56.3.9 qreal Langmuir::SimulationParameters::drainRate

the rate at which all drains accept charges (default, used when eDrainL, etc. are < 0)

6.56.3.10 qreal Langmuir::SimulationParameters::eDrainLRate

the rate at which the left electron drain accepts charges (overrides default)

6.56.3.11 qreal Langmuir::SimulationParameters::eDrainRRate

the rate at which the right electron drain accepts charges (overrides default)

6.56.3.12 qreal Langmuir::SimulationParameters::electronPercentage

the percent of the grid that is reserved for electrons, between 0 and 1

6.56.3.13 qint32 Langmuir::SimulationParameters::electrostaticCutoff

the cut off for Coulomb interations

6.56.3.14 qreal Langmuir::SimulationParameters::electrostaticPrefactor

a compilation of physical constants

6.56.3.15 qreal Langmuir::SimulationParameters::elementaryCharge

physical constant, the elementary charge

6.56.3.16 qreal Langmuir::SimulationParameters::eSourceLRate

the rate at which the left electron source injects charges (overrides default)

6.56.3.17 qreal Langmuir::SimulationParameters::eSourceRRate

the rate at which the right electron source injects charges (overrides default)

6.56.3.18 qreal Langmuir::SimulationParameters::excitonBinding

the energy change (eV) when a hole/electron pair goes from the same site to adjacent sites

6.56.3.19 qreal Langmuir::SimulationParameters::gaussianStdev

the standard deviation of trap sites

6.56.3.20 qreal Langmuir::SimulationParameters::generationRate

the rate at which the exciton source injects charges (overrides default)

6.56.3.21 qreal Langmuir::SimulationParameters::gridFactor

size constant, the size associated with grid sites ($\sim$1nm)

6.56.3.22 qint32 Langmuir::SimulationParameters::gridX

the number of sites along the device length, at least one

6.56.3.23 qint32 Langmuir::SimulationParameters::gridY

the number of sites along the device width, at least one

6.56.3.24 qint32 Langmuir::SimulationParameters::gridZ

the number of sites per layer, at least one

6.56.3.25 qreal Langmuir::SimulationParameters::hDrainLRate

the rate at which the left hole drain accepts charges (overrides default)

6.56.3.26 qreal Langmuir::SimulationParameters::hDrainRRate

the rate at which the right hole drain accepts charges (overrides default)

6.56.3.27 qreal Langmuir::SimulationParameters::holePercentage

the percent of the grid that is reserved for holes, between 0 and 1

6.56.3.28 qint32 Langmuir::SimulationParameters::hoppingRange

the number of sites away from a given site used when calculating neighboring sites

6.56.3.29 qreal Langmuir::SimulationParameters::hSourceLRate

the rate at which the left hole source injects charges (overrides default)

6.56.3.30 qreal Langmuir::SimulationParameters::hSourceRRate

the rate at which the right hole source injects charges (overrides default)

6.56.3.31 qint32 Langmuir::SimulationParameters::imageCarriers

output images of carriers (if n $<$ 0, only at the end; if n == 0, never; if n $>$ 0, every n $*$ iterations.print steps)

6.56.3.32 bool Langmuir::SimulationParameters::imageDefects

output images of defects

6.56.3.33 bool Langmuir::SimulationParameters::imageTraps

output images of defects

6.56.3.34 qreal Langmuir::SimulationParameters::inverseKT

a compilation of physical constants

6.56.3.35 qint32 Langmuir::SimulationParameters::iterationsPrint

if Langmuir, how often to output; if LangmuirView, how many steps between rendering

6.56.3.36 qint32 Langmuir::SimulationParameters::iterationsReal

number of simulation steps after equilibration

6.56.3.37 qint32 Langmuir::SimulationParameters::maxThreads

max threads allowed for QThreadPool - if its $<$= 0 then the QThread::idealThreadCount is used; note that Qt ignores PBS and SGE so when this isn't set Qt will use all the cores on a node

6.56.3.38 bool Langmuir::SimulationParameters::okCL

if true, OpenCL can be used on this platform

6.56.3.39 qint32 Langmuir::SimulationParameters::openclDeviceID

the device to choose if there are multiple

6.56.3.40 qint32 Langmuir::SimulationParameters::openclThreshold

the minimum number of charges that must be present to use OpenCL

6.56.3.41 bool Langmuir::SimulationParameters::outputChkTrapPotential

output trap potentials in checkpoint files

6.56.3.42 qint32 Langmuir::SimulationParameters::outputCoulomb

output coulomb energy for the entire grid (if n $<$ 0, only at the end; if n == 0, never; if n $>$ 0, every n $*$ iterations.print steps)

6.56.3.43 bool Langmuir::SimulationParameters::outputIdsOnDelete

output carrier lifetime and pathlength when they are deleted

6.56.3.44 bool Langmuir::SimulationParameters::outputIdsOnEncounter

output carrier lifetime and pathlength when holes and electrons encounter one another

6.56.3.45 bool Langmuir::SimulationParameters::outputIsOn

if false, produce no output (useful for LangmuirView)

6.56.3.46 bool Langmuir::SimulationParameters::outputPotential

output grid potential at the start of the simulation, includes the trap potential

6.56.3.47 qint32 Langmuir::SimulationParameters::outputPrecision

number of significant figures used for doubles in output

6.56.3.48 qint32 Langmuir::SimulationParameters::outputStepChk

output a checkpoint file every this $*$ iterationsPrint

6.56.3.49 QString Langmuir::SimulationParameters::outputStub

the stub to use when naming output files

6.56.3.50 qint32 Langmuir::SimulationParameters::outputWidth

width of columns in output, ignored in certain files, like trajectory files

6.56.3.51 qint32 Langmuir::SimulationParameters::outputXyz

output trajectory file (if n $<$ 0, only at the end; if n == 0, never; if n $>$ 0, every n $*$ iterations.print steps)

6.56.3.52 bool Langmuir::SimulationParameters::outputXyzD

output defects in trajectory file (ignored if outputXyz is off)

6.56.3.53 bool Langmuir::SimulationParameters::outputXyzE

output electrons in trajectory file (ignored if outputXyz is off)

6.56.3.54 bool Langmuir::SimulationParameters::outputXyzH

output holes in trajectory file (ignored if outputXyz is off)

6.56.3.55 qint32 Langmuir::SimulationParameters::outputXyzMode

output mode for xyz file (if 0, particle count varies; if 1, particle count is constant using "phantom particles")

6.56.3.56 bool Langmuir::SimulationParameters::outputXyzT

output traps in trajectory file (ignored if outputXyz is off)

6.56.3.57 qreal Langmuir::SimulationParameters::permittivitySpace

physical constant, the permittivity of free spece

6.56.3.58 quint64 Langmuir::SimulationParameters::randomSeed

seed the random number generator, if negative, uses the current time (making seperate runs random)

6.56.3.59 qint32 Langmuir::SimulationParameters::recombinationRange

the number of sites to consider when performing recombinations (0 means same-site, 1 means one-site away and same-site)

6.56.3.60 qreal Langmuir::SimulationParameters::recombinationRate

the rate at which holes and electrons can combine when they sit upon one another

6.56.3.61 qreal Langmuir::SimulationParameters::seedCharges

if true, place charges randomly before the simulation starts

6.56.3.62 qreal Langmuir::SimulationParameters::seedPercentage

the percent of the traps to be placed and grown upon to form islands

6.56.3.63 QDateTime Langmuir::SimulationParameters::simulationStart

the time this simulation started

6.56.3.64 QString Langmuir::SimulationParameters::simulationType

tells Langmuir how to set up the Sources and Drains: (`"transistor"`, `"solarcell"`)

6.56.3.65 qreal Langmuir::SimulationParameters::slopeZ

slope of potential along z direction when there are multiple layers (as if there were a gate electrode)

6.56.3.66 bool Langmuir::SimulationParameters::sourceCoulomb

if true, use the Coulomb + Image interaction when calcualting energy change for injection

6.56.3.67 bool Langmuir::SimulationParameters::sourceMetropolis

if true, use an energy change (source potential and site) and metropolis criterion to calculate injection probability for hole and electron sources (not exciton sources)

6.56.3.68 qreal Langmuir::SimulationParameters::sourceRate

the rate at which all sources inject charges

6.56.3.69 qreal Langmuir::SimulationParameters::sourceScaleArea

for SimulationParameters::simulationType == "solarcell", multiply SimulationParameters::sourceRate by (Grid::xy↩PlaneArea)/(SimulationParameters::sourceScaleArea); if <= 0, does not scale rate

6.56.3.70 qreal Langmuir::SimulationParameters::temperatureKelvin

the temperature used in the boltzmann factor

6.56.3.71 qreal Langmuir::SimulationParameters::trapPercentage

the percent of the grid that is reserved for traps, between 0 and 1

6.56.3.72 qreal Langmuir::SimulationParameters::trapPotential

the potential of traps

6.56.3.73 bool Langmuir::SimulationParameters::useOpenCL

if true, try to use OpenCL to speed up Coulomb interaction calculations

6.56.3.74 qreal Langmuir::SimulationParameters::voltageLeft

the potential on the left side of the grid, used in setting up an electric field

6.56.3.75 qreal Langmuir::SimulationParameters::voltageRight

the potential on the right side of the grid, used in setting up an electric field

6.56.3.76 qint32 Langmuir::SimulationParameters::workSize

the size of OpenCL 1DRange kernel work groups

6.56.3.77 qint32 Langmuir::SimulationParameters::workX

the x size of OpenCL 3DRange kernel work groups - only needed if using SimulationParameters::outputCoulomb

6.56.3.78 qint32 Langmuir::SimulationParameters::workY

the y size of OpenCL 3DRange kernel work groups - only needed if using SimulationParameters::outputCoulomb

6.56.3.79 qint32 Langmuir::SimulationParameters::workZ

the z size of OpenCL 3DRange kernel work groups - only needed if using SimulationParameters::outputCoulomb

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

- /home/adam/opt/langmuir/src/langmuirCore/include/parameters.h

6.57 Langmuir::SourceAgent Class Reference

A class to inject charges.

`#include <sourceagent.h>`

Public Member Functions

- SourceAgent (World &world, Grid &grid, QObject *parent=0)
  create a SourceAgent
- bool tryToSeed ()
  seed a charge at a random site
- bool tryToSeed (int site)
  *seed a charge at a **specific** site*
- bool tryToInject ()
  attempt to inject a carrier

Protected Member Functions

- virtual int chooseSite ()
  choose a site to inject to
- virtual bool validToInject (int site)=0
  checks to see if a carrier can actually be injected at the requested site
- virtual void inject (int site)=0
  actually injects carrier.
- virtual bool shouldTransport (int site)
  decides if charge should be injected using a constant probability
- int randomSiteID ()
  choose a random site ID
- int randomNeighborSiteID ()
  choose a random site ID from the neighborlist.

Additional Inherited Members

6.57.1 Detailed Description

A class to inject charges.

6.57.2 Constructor & Destructor Documentation

6.57.2.1 Langmuir::SourceAgent::SourceAgent (World & *world*, Grid & *grid*, QObject * *parent* = 0)

create a SourceAgent

6.57.3 Member Function Documentation

6.57.3.1 virtual int Langmuir::SourceAgent::chooseSite ( ) [protected], [virtual]

choose a site to inject to

By default, choose a site from the SourceAgent's neighborlist. The Grid::CubeFace used to construct the SourceAgent determines the neighborlist.

Reimplemented in Langmuir::ExcitonSourceAgent.

6.57.3.2 virtual void Langmuir::SourceAgent::inject (int *site*) [protected], [pure virtual]

actually injects carrier.

Warning

this function assumes that injecting a charge at the requested site is allowed

Creates a new carrier. Does not perform checks. Forcefully injects charge. Don't call this function unless you know what you are doing.

Implemented in Langmuir::ExcitonSourceAgent, Langmuir::HoleSourceAgent, and Langmuir::ElectronSourceAgent.

6.57.3.3 int Langmuir::SourceAgent::randomNeighborSiteID ( ) [protected]

choose a random site ID from the neighborlist.

6.57.3.4 int Langmuir::SourceAgent::randomSiteID ( ) [protected]

choose a random site ID

It can be any possible site in the grid.

6.57.3.5 virtual bool Langmuir::SourceAgent::shouldTransport (int *site*) [protected], [virtual]

decides if charge should be injected using a constant probability

Parameters

site	the site involved

If SimulationParameters::sourceMetropolis is true, then use the metropolis criterion with an energy change to decide if charge should be injected.

Reimplemented from Langmuir::FluxAgent.

Reimplemented in Langmuir::ExcitonSourceAgent.

6.57.3.6 bool Langmuir::SourceAgent::tryToInject ( )

attempt to inject a carrier

This is the main transport method of a SourceAgent. This function uses chooseSite(), shouldTransport() and validToInject() to inject the charge. It is not garunteed that a charge will be injected.

6.57.3.7 bool Langmuir::SourceAgent::tryToSeed ( )

seed a charge at a random site

Warning

does not call shouldTransport()

Attempts to seed a charge at a random site, without calling shouldTransport(). However, validToInject() is still called. This function is used when randomly placing charges in the system.

6.57.3.8 bool Langmuir::SourceAgent::tryToSeed (int *site*)

seed a charge at a **specific** site

Warning

does not call shouldTransport()

Attempts to seed a charge at a specific site, without calling shouldTransport(). However, validToInject() is still called. This function is used when placing charges at specific places. For example, when sometimes the checkpoint file has information on where charges are/were, and these need to be placed.

6.57.3.9 virtual bool Langmuir::SourceAgent::validToInject (int *site*) [protected], [pure virtual]

checks to see if a carrier can actually be injected at the requested site

For example, if the site contains a defect, or a carrier is already present at the site, then it is not valid to inject the carrier at this site.

Implemented in Langmuir::ExcitonSourceAgent, Langmuir::HoleSourceAgent, and Langmuir::ElectronSourceAgent.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/sourceagent.h

6.58 Langmuir::SSpinBox Class Reference

`#include <gridview.h>`

Public Slots

- void setValueSlot (int value)

Public Member Functions

- SSpinBox (QWidget *parent)

6.58.1 Constructor & Destructor Documentation

6.58.1.1 Langmuir::SSpinBox::SSpinBox (QWidget * *parent*) [inline]

6.58.2 Member Function Documentation

6.58.2.1 void Langmuir::SSpinBox::setValueSlot (int *value*) [slot]

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

- /home/adam/opt/langmuir/src/langmuirView/include/gridview.h

6.59 MarchingCubes::Triangle Class Reference

Container for vertices and normals of triangle.

`#include <isosurface.h>`

Public Member Functions

- Triangle (QObject *parent=0)

  create a Triangle
- ~Triangle ()
- void setVertex (int index, QVector3D vector)

  set a vertex by index
- void calculateNormals ()

  calculate normal vectors
- void sort ()

  sort vertices

Public Attributes

- QVector3D v0

  vertex 0
- QVector3D v1

  vertex 1
- QVector3D v2

  vertex 2
- QVector3D n0

  normal 0
- QVector3D n1

  normal 1
- QVector3D n2

  normal 2

6.59.1 Detailed Description

Container for vertices and normals of triangle.

6.59.2 Constructor & Destructor Documentation

6.59.2.1 MarchingCubes::Triangle::Triangle (QObject * *parent* = 0) [explicit]

create a Triangle

6.59.2.2 MarchingCubes::Triangle::~Triangle ( )

6.59.3 Member Function Documentation

6.59.3.1 void MarchingCubes::Triangle::calculateNormals ( )

calculate normal vectors

6.59.3.2 void MarchingCubes::Triangle::setVertex (int *index*, QVector3D *vector*)

set a vertex by index

Parameters

index	index of vertex
vector	vector to set

6.59.3.3 void MarchingCubes::Triangle::sort ( )

sort vertices

6.59.4 Member Data Documentation

6.59.4.1 QVector3D MarchingCubes::Triangle::n0

normal 0

6.59.4.2 QVector3D MarchingCubes::Triangle::n1

normal 1

6.59.4.3 QVector3D MarchingCubes::Triangle::n2

normal 2

6.59.4.4 QVector3D MarchingCubes::Triangle::v0

vertex 0

6.59.4.5 QVector3D MarchingCubes::Triangle::v1

vertex 1

6.59.4.6 QVector3D MarchingCubes::Triangle::v2

vertex 2

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

- /home/adam/opt/langmuir/src/langmuirView/include/isosurface.h

6.60 Langmuir::TypedVariable< T > Class Template Reference

A template class to map between variable names (keys) and locations (references)

`#include <variable.h>`

Public Member Functions

- TypedVariable (const QString &key, T &value, VariableMode mode=0, QObject *parent=0)
  Create a new variable.
- virtual void read (const QString &token)
  Cast the value stored in string to the correct type and store it in the correct location.
- virtual QString key () const
  Get this variable's key (name)
- virtual QString value () const
  Get this variable's value as a QString.
- virtual QString keyValue () const
  Get this variable's key and value in the form 'key = value'.
- template<>
  QString value () const
  Get this variable's value as a QString.
- template<>
  QString value () const
  Get this variable's value as a QString.
- template<>
  QString value () const
  Get this variable's value as a QString.
- template<>
  QString keyValue () const
  Get this variable's key and value in the form 'key = value'.
- template<>
  void convert (const QString &token, QString &result)
- template<>
  void convert (const QString &token, qreal &result)
- template<>
  void convert (const QString &token, float &result)
- template<>
  void convert (const QString &token, bool &result)
- template<>
  void convert (const QString &token, qint32 &result)
- template<>
  void convert (const QString &token, quint32 &result)
- template<>
  void convert (const QString &token, qint64 &result)
- template<>
  void convert (const QString &token, quint64 &result)
- template<>
  void convert (const QString &token, QDateTime &result)

Static Public Member Functions

- static void convert (const QString &token, T &result)

  A template function for converting a QString to some type T.

Protected Member Functions

- virtual void write (QTextStream &stream) const

  Write 'key = value' to a stream.

Protected Attributes

- T & m_value

  Reference to the object being tracked.

Additional Inherited Members

6.60.1 Detailed Description

template<class T>class Langmuir::TypedVariable< T >

A template class to map between variable names (keys) and locations (references)

6.60.2 Constructor & Destructor Documentation

6.60.2.1 template<class T > Langmuir::TypedVariable< T >::TypedVariable (const QString & *key*, T & *value*, VariableMode *mode* = 0, QObject * *parent* = 0) [inline]

Create a new variable.

initialize Variable with a key and a location

Parameters

key	the name of the variable
value	the location where the value of the variable is stored
mode	flags to alter variable's behavoir, see AbstractVariable::VariableModeFlag
parent	QObject this belongs to

6.60.3 Member Function Documentation

6.60.3.1 template<class T > static void Langmuir::TypedVariable< T >::convert (const QString & *token*, T & *result*) [static]

A template function for converting a QString to some type T.

To implement this function for a new data type, use declarations of the form:

- template <> inline void Variable<T>::convert(const QString& token, T& result)
- replace the **'T'** with the data type you want to implement (for example, double).

6.60.3.2 template<> void Langmuir::TypedVariable< QString >::convert (const QString & *token,* QString & *result*) [inline]

6.60.3.3 template<> void Langmuir::TypedVariable< qreal >::convert (const QString & *token,* qreal & *result*) [inline]

6.60.3.4 template<> void Langmuir::TypedVariable< float >::convert (const QString & *token,* float & *result*) [inline]

6.60.3.5 template<> void Langmuir::TypedVariable< bool >::convert (const QString & *token,* bool & *result*) [inline]

6.60.3.6 template<> void Langmuir::TypedVariable< qint32 >::convert (const QString & *token,* qint32 & *result*) [inline]

6.60.3.7 template<> void Langmuir::TypedVariable< quint32 >::convert (const QString & *token,* quint32 & *result*) [inline]

6.60.3.8 template<> void Langmuir::TypedVariable< qint64 >::convert (const QString & *token,* qint64 & *result*) [inline]

6.60.3.9 template<> void Langmuir::TypedVariable< quint64 >::convert (const QString & *token,* quint64 & *result*) [inline]

6.60.3.10 template<> void Langmuir::TypedVariable< QDateTime >::convert (const QString & *token,* QDateTime & *result*) [inline]

6.60.3.11 template<class T > QString Langmuir::TypedVariable< T >::key ( ) const [inline], [virtual]

Get this variable's key (name)

get the variable's key

Implements Langmuir::Variable.

6.60.3.12 template<class T > QString Langmuir::TypedVariable< T >::keyValue ( ) const [inline], [virtual]

Get this variable's key and value in the form 'key = value'.

get the variable's key

Implements Langmuir::Variable.

6.60.3.13 template<> QString Langmuir::TypedVariable< QString >::keyValue ( ) const [inline], [virtual]

Get this variable's key and value in the form 'key = value'.

Implements Langmuir::Variable.

6.60.3.14 template<class T > void Langmuir::TypedVariable< T >::read (const QString & *token*) [inline], [virtual]

Cast the value stored in string to the correct type and store it in the correct location.

convert a QString token to its correct type

Implements Langmuir::Variable.

6.60.3.15 template<class T > QString Langmuir::TypedVariable< T >::value ( ) const [inline], [virtual]

Get this variable's value as a QString.

get the variable's value (converted to string)

Implements Langmuir::Variable.

6.60.3.16 template<> QString Langmuir::TypedVariable< float >::value ( ) const [inline], [virtual]

Get this variable's value as a QString.

Implements Langmuir::Variable.

6.60.3.17 template<> QString Langmuir::TypedVariable< qreal >::value ( ) const [inline], [virtual]

Get this variable's value as a QString.

Implements Langmuir::Variable.

6.60.3.18 template<> QString Langmuir::TypedVariable< bool >::value ( ) const [inline], [virtual]

Get this variable's value as a QString.

Implements Langmuir::Variable.

6.60.3.19 template<class T > void Langmuir::TypedVariable< T >::write (QTextStream & *stream*) const [inline], [protected], [virtual]

Write 'key = value' to a stream.

write keyValue() to a stream

Implements Langmuir::Variable.

6.60.4 Member Data Documentation

6.60.4.1 template<class T > T& Langmuir::TypedVariable< T >::m_value [protected]

Reference to the object being tracked.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/variable.h

6.61 Langmuir::Variable Class Reference

A class to map between variable names (keys) and locations (references)

`#include <variable.h>`

Public Types

- enum VariableModeFlag { Constant = 1 }

  A Flag to alter the behavoir of certain variable member functions.

Public Member Functions

- Variable (const QString &key, VariableMode mode=0, QObject *parent=0)

  Create a Variable, see Variable::Variable for description.
- virtual void read (const QString &token)=0

  Cast the value stored in string to the correct type and store it in the correct location.
- virtual QString key () const =0

  Get this variable's key (name)
- virtual QString value () const =0

  Get this variable's value as a QString.
- virtual QString keyValue () const =0

  Get this variable's key and value in the form 'key = value'.
- bool isConstant () const

  True if the Variable::Constant mode flag was set.
- const VariableMode & mode () const

  Get this variable's mode flags.

Protected Member Functions

- virtual void write (QTextStream &stream) const =0

  Write 'key = value' to a stream.

Protected Attributes

- QString m_key

  The name of this variable.
- VariableMode m_mode

  The mode flags for this variable.

Friends

- QTextStream & operator<< (QTextStream &stream, const Variable &variable)

  Operator overload to output 'key = value' to QTextStream.
- QDebug operator<< (QDebug dbg, const Variable &variable)

  Operator overload to output 'key = value' to QDebug.
- std::ostream & operator<< (std::ostream &stream, Variable &variable)

  Operator overload to output to output 'key = value' to std::ofstream.

6.61.1 Detailed Description

A class to map between variable names (keys) and locations (references)

6.61.2 Member Enumeration Documentation

6.61.2.1 enum Langmuir::Variable::VariableModeFlag

A Flag to alter the behavoir of certain variable member functions.

Enumerator

Constant When constant, a variable's read / convert function does nothing.

6.61.3 Constructor & Destructor Documentation

6.61.3.1 Langmuir::Variable::Variable (const QString & *key*, VariableMode *mode* = 0, QObject * *parent* = 0) [inline]

Create a Variable, see Variable::Variable for description.

initialize a Variable with a key

6.61.4 Member Function Documentation

6.61.4.1 bool Langmuir::Variable::isConstant ( ) const [inline]

True if the Variable::Constant mode flag was set.

see if the Variable is really a Constant

6.61.4.2 virtual QString Langmuir::Variable::key ( ) const [pure virtual]

Get this variable's key (name)

Implemented in Langmuir::TypedVariable< T >.

6.61.4.3 virtual QString Langmuir::Variable::keyValue ( ) const [pure virtual]

Get this variable's key and value in the form 'key = value'.

Implemented in Langmuir::TypedVariable< T >, and Langmuir::TypedVariable< T >.

6.61.4.4 const Variable::VariableMode & Langmuir::Variable::mode ( ) const [inline]

Get this variable's mode flags.

get the mode flags of this Variable

6.61.4.5 virtual void Langmuir::Variable::read (const QString & *token*) [pure virtual]

Cast the value stored in string to the correct type and store it in the correct location.

This function assumes **'QTextStream&** operator<<' has been implemented for that data type T. Keep this in mind if adding a new data type.

Implemented in Langmuir::TypedVariable< T >.

6.61.4.6 virtual QString Langmuir::Variable::value ( ) const [pure virtual]

Get this variable's value as a QString.

Implemented in Langmuir::TypedVariable< T >, Langmuir::TypedVariable< T >, Langmuir::TypedVariable< T >, and Langmuir::TypedVariable< T >.

6.61.4.7 virtual void Langmuir::Variable::write (QTextStream & *stream*) const [protected], [pure virtual]

Write 'key = value' to a stream.

Implemented in Langmuir::TypedVariable< T >.

6.61.5 Friends And Related Function Documentation

6.61.5.1 QTextStream& operator<< (QTextStream & *stream,* const Variable & *variable*) [friend]

Operator overload to output 'key = value' to QTextStream.

6.61.5.2 QDebug operator<< (QDebug *dbg,* const Variable & *variable*) [friend]

Operator overload to output 'key = value' to QDebug.

6.61.5.3 std::ostream& operator<< (std::ostream & *stream,* Variable & *variable*) [friend]

Operator overload to output to output 'key = value' to std::ofstream.

6.61.6 Member Data Documentation

6.61.6.1 QString Langmuir::Variable::m_key [protected]

The name of this variable.

6.61.6.2 VariableMode Langmuir::Variable::m_mode [protected]

The mode flags for this variable.

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

- /home/adam/opt/langmuir/src/langmuirCore/include/variable.h

6.62 Langmuir::World Class Reference

A class to hold all objects in a simulation.

`#include <world.h>`

Public Member Functions

- World (const QString &fileName, int cores=-1, int gpuID=-1, QObject *parent=0)
  create a world to simulate in
- World (SimulationParameters &parameters, int cores=-1, int gpuID=-1, QObject *parent=0)
- World (SimulationParameters &parameters, ConfigurationInfo &configInfo, int cores=-1, int gpuID=-1, QObject *parent=0)
- ~World ()
  destroys the entire World, and everything in it...including you.
- KeyValueParser & keyValueParser ()
  get the KeyValueParser, used for parsing input files.
- CheckPointer & checkPointer ()
  get the CheckPointer, used for reading and writing input files.
- Grid & electronGrid ()
  get the Grid used, used for holding ElectronAgents
- Grid & holeGrid ()
  get the hole Grid, used for holding HoleAgents

- Potential & potential ()

  get the Potential, a calculator used for...calculating the potential.
- SimulationParameters & parameters ()

  get the SimulationParameters, a struct used for holding simulation parameters.
- Random & randomNumberGenerator ()

  get the Random, used for creating random numbers
- Logger & logger ()

  get the Logger, used for writing output
- OpenClHelper & opencl ()

  get the OpenClHelper, used for calculating Coulomb interactions with a Graphics Card
- QList< SourceAgent * > & sources ()

  get a list of all SourceAgents
- QList< SourceAgent * > & eSources ()

  get a list of all ElectronSourceAgents
- QList< SourceAgent * > & hSources ()

  get a list of all ElectronSourceAgents
- QList< SourceAgent * > & xSources ()

  get a list of all ElectronSourceAgents
- QList< DrainAgent * > & drains ()

  get a list of all DrainAgents
- QList< DrainAgent * > & eDrains ()

  get a list of all ElectronSourceAgents
- QList< DrainAgent * > & hDrains ()

  get a list of all ElectronSourceAgents
- QList< DrainAgent * > & xDrains ()

  get a list of all ElectronSourceAgents
- QList< FluxAgent * > & fluxes ()

  get a list of all FluxAgents
- ElectronSourceAgent & electronSourceAgentRight ()

  get the right ElectronSourceAgent
- ElectronSourceAgent & electronSourceAgentLeft ()

  get the left ElectronSourceAgent
- HoleSourceAgent & holeSourceAgentRight ()

  get the right HoleSourceAgent
- HoleSourceAgent & holeSourceAgentLeft ()

  get the left HoleSourceAgent
- ExcitonSourceAgent & excitonSourceAgent ()

  get the RecombinationAgent
- ElectronDrainAgent & electronDrainAgentRight ()

  get the right ElectronDrainAgent
- ElectronDrainAgent & electronDrainAgentLeft ()

  get the left ElectronDrainAgent
- HoleDrainAgent & holeDrainAgentRight ()

  get the right HoleDrainAgent
- HoleDrainAgent & holeDrainAgentLeft ()

  get the left HoleDrainAgent
- RecombinationAgent & recombinationAgent ()

  get the RecombinationAgent
- QList< ChargeAgent * > & electrons ()

  get a list of all ElectronAgents
- QList< ChargeAgent * > & holes ()

get a list of all HoleAgents

- QList< int > & defectSiteIDs ()

  get a list of all defect sites
- QList< int > & trapSiteIDs ()

  get a list of all trap sites
- QList< double > & trapSitePotentials ()

  get a list of all trap potentials
- boost::multi_array< double, 3 > & R1 ()

  get the array of precomputed r-squared values
- boost::multi_array< double, 3 > & R2 ()

  get the array of precomputed r values
- boost::multi_array< double, 3 > & iR ()

  get the array of precomputed inverse-r values
- boost::multi_array< double, 3 > & eR ()

  get the array of precomputed erf(r/(sqrt(2)∗sigma))
- boost::multi_array< double, 3 > & sI ()

  get the self interactions
- boost::multi_array< double, 3 > & couplingConstants ()

  get the coupling constants
- int maxElectronAgents ()

  get the max number of ElectronAgents allowed
- int maxHoleAgents ()

  get the max number of HoleAgents allowed
- int maxChargeAgents ()

  get the max number of ChargeAgents allowed
- int maxChargeAgentsAndChargedDefects ()

  get the max number of ChargeAgents & charged defects
- int maxDefects ()

  get the max number of Defects allowed
- int maxTraps ()

  get the max number of Traps allowed
- int numElectronAgents ()

  get the current number of ElectronAgents
- int numHoleAgents ()

  get the current number of HoleAgents
- int numChargeAgents ()

  get the current number of ChargeAgents
- int electronsMinusHoles ()

  The number of electrons - holes.
- int holesMinusElectrons ()

  The number of holes - electrons.
- bool chargesAreBalanced ()

  true when electrons and holes are balanced
- int numChargeAgentsAndChargedDefects ()

  get the current number of ChargeAgents & charged defects
- int numDefects ()

  get the current number of Defects
- int numTraps ()

  get the current number of Traps
- double reachedChargeAgents ()

  get the percent of ChargeAgents reached, of the max allowed

- double reachedElectronAgents ()

  get the percent of ElectronAgents reached, of the max allowed
- double reachedHoleAgents ()

  get the percent of HoleAgents reached, of the max allowed
- double percentHoleAgents ()

  get the percent of HoleAgents reached, of the total grid volume
- double percentElectronAgents ()

  get the percent of ElectronAgents reached, of the total grid volume
- bool atMaxElectrons ()

  check if the maximum number of electrons has been reached
- bool atMaxHoles ()

  check if the maximum number of holes has been reached
- bool atMaxCharges ()

  check if the maximum number of charges has been reached

Private Member Functions

- void placeDefects (const QList< int > &siteIDs=QList< int >())

  places defects
- void placeElectrons (const QList< int > &siteIDs=QList< int >())

  places electrons
- void placeHoles (const QList< int > &siteIDs=QList< int >())

  places holes
- void createSources ()

  create SourceAgents
- void createDrains ()

  create DrainAgents
- void setFluxInfo (const QList< quint64 > &fluxInfo)

  set attempts / successes for sources / drains
- void alterMaxThreads (int cores=-1)

  Change the number of cores used.
- void initialize (const QString &fileName="", SimulationParameters *pparameters=NULL, ConfigurationInfo *pconfigInfo=NULL, int cores=-1, int gpuID=-1)

  initialize all objects

Private Attributes

- KeyValueParser * m_keyValueParser

  pointer to KeyValueParser, used for parsing key=value pairs
- CheckPointer * m_checkPointer

  pointer to CheckPointer, used for reading/writing input(checkpoint) files
- QList< SourceAgent * > m_sources

  list of SourceAgents
- QList< SourceAgent * > m_eSources

  list of ElectronSourceAgents
- QList< SourceAgent * > m_hSources

  list of HoleSourceAgents
- QList< SourceAgent * > m_xSources

  list of ExcitonSourceAgents
- QList< DrainAgent * > m_drains

list of DrainAgents

- QList< DrainAgent * > m_eDrains

  list of ElectronSourceAgents
- QList< DrainAgent * > m_hDrains

  list of HoleSourceAgents
- QList< DrainAgent * > m_xDrains

  list of ExcitonSourceAgents
- QList< FluxAgent * > m_fluxAgents

  list of all FluxAgents, such as SoureAgents, DrainAgents, etc.
- ElectronSourceAgent * m_electronSourceAgentRight

  pointer to right ElectronDrainAgent
- ElectronSourceAgent * m_electronSourceAgentLeft

  pointer to left ElectronDrainAgent
- HoleSourceAgent * m_holeSourceAgentRight

  pointer to right HoleDrainAgent
- HoleSourceAgent * m_holeSourceAgentLeft

  pointer to left HoleDrainAgent
- ExcitonSourceAgent * m_excitonSourceAgent

  pointer to ExcitonSourceAgent, used for injecting Excitons
- ElectronDrainAgent * m_electronDrainAgentRight

  pointer to right ElectronDrainAgent
- ElectronDrainAgent * m_electronDrainAgentLeft

  pointer to left ElectronDrainAgent
- HoleDrainAgent * m_holeDrainAgentRight

  pointer to right HoleDrainAgent
- HoleDrainAgent * m_holeDrainAgentLeft

  pointer to left HoleDrainAgent
- RecombinationAgent * m_recombinationAgent

  pointer to electron/hole RecombinationAgent, used for removing Excitons
- Grid * m_electronGrid

  pointer to electron Grid, used for keeping track of ElectronAgents
- Grid * m_holeGrid

  pointer to hole Grid, used for keeping track of HoleAgents
- Random * m_rand

  pointer to Random, used for generating random numbers
- Potential * m_potential

  pointer to Potential, used for calculating the potential
- SimulationParameters * m_parameters

  pointer to SimulationParameters
- Logger * m_logger

  pointer to Logger, used for output
- OpenCLHelper * m_ocl

  pointer to OpenCLHelper, used for Graphics Card calculations
- QList< ChargeAgent * > m_electrons

  list of electrons
- QList< ChargeAgent * > m_holes

  list of holes
- QList< int > m_defectSiteIDs

  list of defect sites
- QList< int > m_trapSiteIDs

  list of trap sites

- QList< double > m_trapSitePotentials

  list of trap potentials
- boost::multi_array< double, 3 > m_R2

  array of precomputed r-squared values
- boost::multi_array< double, 3 > m_R1

  array of precomputed r values
- boost::multi_array< double, 3 > m_iR

  array of precomputed inverse-r values
- boost::multi_array< double, 3 > m_eR

  *array of precomputed erf(r/(s*sqrt(2)) values*
- boost::multi_array< double, 3 > m_sI

  self interaction, which is 1/(4 pi e e0 r), with r=1 grid unit When a charge at it's future site interacts with other charges at their current site, the charge will interact with it's own current site. So, this value needs to be subtracted off.
- boost::multi_array< double, 3 > m_couplingConstants

  array of coupling constants
- int m_maxElectrons

  max number of electrons
- int m_maxHoles

  max number of holes
- int m_maxDefects

  max number of defects
- int m_maxTraps

  max number of traps

6.62.1 Detailed Description

A class to hold all objects in a simulation.

6.62.2 Constructor & Destructor Documentation

6.62.2.1 Langmuir::World::World (const QString & *fileName*, int *cores* = -1, int *gpuID* = -1, QObject * *parent* = 0)

create a world to simulate in

Parameters

fileName	the input file name
parent	QObject this belongs to

Calls the initialize() function.

6.62.2.2 Langmuir::World::World (SimulationParameters & *parameters*, int *cores* = -1, int *gpuID* = -1, QObject * *parent* = 0)

6.62.2.3 Langmuir::World::World (SimulationParameters & *parameters*, ConfigurationInfo & *configInfo*, int *cores* = -1, int *gpuID* = -1, QObject * *parent* = 0)

6.62.2.4 Langmuir::World::~World ( )

destroys the entire World, and everything in it...including you.

6.62.3 Member Function Documentation

6.62.3.1 void Langmuir::World::alterMaxThreads (int *cores* = -1) [private]

Change the number of cores used.

Parameters

cores	the number of cores

6.62.3.2 bool Langmuir::World::atMaxCharges ( )

check if the maximum number of charges has been reached

6.62.3.3 bool Langmuir::World::atMaxElectrons ( )

check if the maximum number of electrons has been reached

6.62.3.4 bool Langmuir::World::atMaxHoles ( )

check if the maximum number of holes has been reached

6.62.3.5 bool Langmuir::World::chargesAreBalanced ( )

true when electrons and holes are balanced

6.62.3.6 CheckPointer& Langmuir::World::checkPointer ( )

get the CheckPointer, used for reading and writing input files.

6.62.3.7 boost::multi_array<double,3>& Langmuir::World::couplingConstants ( )

get the coupling constants

6.62.3.8 void Langmuir::World::createDrains ( ) [private]

create DrainAgents

6.62.3.9 void Langmuir::World::createSources ( ) [private]

create SourceAgents

6.62.3.10 QList<int>& Langmuir::World::defectSiteIDs ( )

get a list of all defect sites

6.62.3.11 QList<DrainAgent*>& Langmuir::World::drains ( )

get a list of all DrainAgents

6.62.3.12 QList<DrainAgent*>& Langmuir::World::eDrains ( )

get a list of all ElectronSourceAgents

6.62.3.13 ElectronDrainAgent& Langmuir::World::electronDrainAgentLeft ( )

get the left ElectronDrainAgent

6.62.3.14 ElectronDrainAgent& Langmuir::World::electronDrainAgentRight ( )

get the right ElectronDrainAgent

6.62.3.15 Grid& Langmuir::World::electronGrid ( )

get the Grid used, used for holding ElectronAgents

6.62.3.16 QList<ChargeAgent*>& Langmuir::World::electrons ( )

get a list of all ElectronAgents

6.62.3.17 int Langmuir::World::electronsMinusHoles ( )

The number of electrons - holes.

6.62.3.18 ElectronSourceAgent& Langmuir::World::electronSourceAgentLeft ( )

get the left ElectronSourceAgent

6.62.3.19 ElectronSourceAgent& Langmuir::World::electronSourceAgentRight ( )

get the right ElectronSourceAgent

6.62.3.20 boost::multi_array<double,3>& Langmuir::World::eR ( )

get the array of precomputed erf(r/(sqrt(2)*sigma))

6.62.3.21 QList<SourceAgent*>& Langmuir::World::eSources ( )

get a list of all ElectronSourceAgents

6.62.3.22 ExcitonSourceAgent& Langmuir::World::excitonSourceAgent ( )

get the RecombinationAgent

6.62.3.23 QList<FluxAgent*>& Langmuir::World::fluxes ( )

get a list of all FluxAgents

6.62.3.24 QList<DrainAgent*>& Langmuir::World::hDrains ( )

get a list of all ElectronSourceAgents

6.62.3.25 HoleDrainAgent& Langmuir::World::holeDrainAgentLeft ( )

get the left HoleDrainAgent

6.62.3.26 HoleDrainAgent& Langmuir::World::holeDrainAgentRight ( )

get the right HoleDrainAgent

6.62.3.27 Grid& Langmuir::World::holeGrid ( )

get the hole Grid, used for holding HoleAgents

6.62.3.28 QList<ChargeAgent*>& Langmuir::World::holes ( )

get a list of all HoleAgents

6.62.3.29 int Langmuir::World::holesMinusElectrons ( )

The number of holes - electrons.

6.62.3.30 HoleSourceAgent& Langmuir::World::holeSourceAgentLeft ( )

get the left HoleSourceAgent

6.62.3.31 HoleSourceAgent& Langmuir::World::holeSourceAgentRight ( )

get the right HoleSourceAgent

6.62.3.32 QList<SourceAgent*>& Langmuir::World::hSources ( )

get a list of all ElectronSourceAgents

6.62.3.33 void Langmuir::World::initialize (const QString & *fileName* = " ", SimulationParameters * *pparameters* = NULL, ConfigurationInfo * *pconfigInfo* = NULL, int *cores* = -1, int *gpuID* = -1) [private]

initialize all objects

Parameters

fileName	input file name

A very long, though not all that complicated function that creates all the simulation objects. Best to read through it in the source code.

6.62.3.34 boost::multi_array<double,3>& Langmuir::World::iR ( )

get the array of precomputed inverse-r values

6.62.3.35 KeyValueParser& Langmuir::World::keyValueParser ( )

get the KeyValueParser, used for parsing input files.

6.62.3.36 Logger& Langmuir::World::logger ( )

get the Logger, used for writing output

6.62.3.37 int Langmuir::World::maxChargeAgents ( )

get the max number of ChargeAgents allowed

6.62.3.38 int Langmuir::World::maxChargeAgentsAndChargedDefects ( )

get the max number of ChargeAgents & charged defects

6.62.3.39 int Langmuir::World::maxDefects ( )

get the max number of Defects allowed

6.62.3.40 int Langmuir::World::maxElectronAgents ( )

get the max number of ElectronAgents allowed

6.62.3.41 int Langmuir::World::maxHoleAgents ( )

get the max number of HoleAgents allowed

6.62.3.42 int Langmuir::World::maxTraps ( )

get the max number of Traps allowed

6.62.3.43 int Langmuir::World::numChargeAgents ( )

get the current number of ChargeAgents

6.62.3.44 int Langmuir::World::numChargeAgentsAndChargedDefects ( )

get the current number of ChargeAgents & charged defects

6.62.3.45 int Langmuir::World::numDefects ( )

get the current number of Defects

6.62.3.46 int Langmuir::World::numElectronAgents ( )

get the current number of ElectronAgents

6.62.3.47 int Langmuir::World::numHoleAgents ( )

get the current number of HoleAgents

6.62.3.48 int Langmuir::World::numTraps ( )

get the current number of Traps

6.62.3.49 OpenCLHelper& Langmuir::World::opencl ( )

get the OpenCLHelper, used for calculating Coulomb interactions with a Graphics Card

6.62.3.50 SimulationParameters& Langmuir::World::parameters ( )

get the SimulationParameters, a struct used for holding simulation parameters.

6.62.3.51 double Langmuir::World::percentElectronAgents ( )

get the percent of ElectronAgents reached, of the total grid volume

6.62.3.52 double Langmuir::World::percentHoleAgents ( )

get the percent of HoleAgents reached, of the total grid volume

6.62.3.53 void Langmuir::World::placeDefects (const QList< int > & *siteIDs* = QList< int >()) [private]

places defects

Parameters

siteIDs	a list of defect site ids

Places carriers according to the site ids passed. If more need placing (according to SimulationParameters::seedCharges), then they are placed randomly.

6.62.3.54 void Langmuir::World::placeElectrons (const QList< int > & *siteIDs* = QList< int >()) [private]

places electrons

Parameters

siteIDs	a list of electron site ids

Places carriers according to the site ids passed. If more need placing (according to SimulationParameters::seedCharges), then they are placed randomly.

6.62.3.55 void Langmuir::World::placeHoles (const QList< int > & *siteIDs* = QList< int >()) [private]

places holes

Parameters

siteIDs	a list of hole site ids

Places carriers according to the site ids passed. If more need placing (according to SimulationParameters::seedCharges), then they are placed randomly.

6.62.3.56 Potential& Langmuir::World::potential ( )

get the Potential, a calculator used for...calculating the potential.

6.62.3.57 boost::multi_array<double,3>& Langmuir::World::R1 ( )

get the array of precomputed r-squared values

6.62.3.58 boost::multi_array<double,3>& Langmuir::World::R2 ( )

get the array of precomputed r values

6.62.3.59 Random& Langmuir::World::randomNumberGenerator ( )

get the Random, used for creating random numbers

6.62.3.60 double Langmuir::World::reachedChargeAgents ( )

get the percent of ChargeAgents reached, of the max allowed

6.62.3.61 double Langmuir::World::reachedElectronAgents ( )

get the percent of ElectronAgents reached, of the max allowed

6.62.3.62 double Langmuir::World::reachedHoleAgents ( )

get the percent of HoleAgents reached, of the max allowed

6.62.3.63 RecombinationAgent& Langmuir::World::recombinationAgent ( )

get the RecombinationAgent

6.62.3.64 void Langmuir::World::setFluxInfo (const QList< quint64 > & *fluxInfo*) [private]

set attempts / successes for sources / drains

6.62.3.65 boost::multi_array<double, 3>& Langmuir::World::sI ( )

get the self interactions

6.62.3.66 QList<SourceAgent*>& Langmuir::World::sources ( )

get a list of all SourceAgents

6.62.3.67 QList<int>& Langmuir::World::trapSiteIDs ( )

get a list of all trap sites

6.62.3.68 QList<double>& Langmuir::World::trapSitePotentials ( )

get a list of all trap potentials

6.62.3.69 QList<DrainAgent*>& Langmuir::World::xDrains ( )

get a list of all ElectronSourceAgents

6.62.3.70 QList<SourceAgent*>& Langmuir::World::xSources ( )

get a list of all ElectronSourceAgents

6.62.4 Member Data Documentation

6.62.4.1 CheckPointer* Langmuir::World::m_checkPointer [private]

pointer to CheckPointer, used for reading/writing input(checkpoint) files

6.62.4.2 boost::multi_array<double,3> Langmuir::World::m_couplingConstants [private]

array of coupling constants
This array is indexed by dx, dy, dz values.

6.62.4.3 QList<int> Langmuir::World::m_defectSiteIDs [private]

list of defect sites

6.62.4.4 QList<DrainAgent*> Langmuir::World::m_drains [private]

list of DrainAgents

6.62.4.5 QList<DrainAgent*> Langmuir::World::m_eDrains [private]

list of ElectronSourceAgents

6.62.4.6 ElectronDrainAgent* Langmuir::World::m_electronDrainAgentLeft [private]

pointer to left ElectronDrainAgent

6.62.4.7 ElectronDrainAgent* Langmuir::World::m_electronDrainAgentRight [private]

pointer to right ElectronDrainAgent

6.62.4.8 Grid* Langmuir::World::m_electronGrid [private]

pointer to electron Grid, used for keeping track of ElectronAgents

6.62.4.9 QList<ChargeAgent*> Langmuir::World::m_electrons [private]

list of electrons

6.62.4.10 ElectronSourceAgent* Langmuir::World::m_electronSourceAgentLeft [private]

pointer to left ElectronDrainAgent

6.62.4.11 ElectronSourceAgent* Langmuir::World::m_electronSourceAgentRight [private]

pointer to right ElectronDrainAgent

6.62.4.12 boost::multi_array<double,3> Langmuir::World::m_eR [private]

array of precomputed erf(r/(s*sqrt(2)) values
This array is indexed by dx, dy, dz values, and r is in grid-units

6.62.4.13 QList<SourceAgent*> Langmuir::World::m_eSources [private]

list of ElectronSourceAgents

6.62.4.14 ExcitonSourceAgent* Langmuir::World::m_excitonSourceAgent [private]

pointer to ExcitonSourceAgent, used for injecting Excitons

6.62.4.15 QList<FluxAgent*> Langmuir::World::m_fluxAgents [private]

list of all FluxAgents, such as SoureAgents, DrainAgents, etc.

6.62.4.16 QList<DrainAgent*> Langmuir::World::m_hDrains [private]

list of HoleSourceAgents

6.62.4.17 HoleDrainAgent* Langmuir::World::m_holeDrainAgentLeft [private]

pointer to left HoleDrainAgent

6.62.4.18 HoleDrainAgent* Langmuir::World::m_holeDrainAgentRight [private]

pointer to right HoleDrainAgent

6.62.4.19 Grid* Langmuir::World::m_holeGrid [private]

pointer to hole Grid, used for keeping track of HoleAgents

6.62.4.20 QList<ChargeAgent*> Langmuir::World::m_holes [private]

list of holes

6.62.4.21 HoleSourceAgent* Langmuir::World::m_holeSourceAgentLeft [private]

pointer to left HoleDrainAgent

6.62.4.22 HoleSourceAgent* Langmuir::World::m_holeSourceAgentRight [private]

pointer to right HoleDrainAgent

6.62.4.23 QList<SourceAgent*> Langmuir::World::m_hSources [private]

list of HoleSourceAgents

6.62.4.24 boost::multi_array<double,3> Langmuir::World::m_iR [private]

array of precomputed inverse-r values
This array is indexed by dx, dy, dz values, and r is in grid-units

6.62.4.25 KeyValueParser* Langmuir::World::m_keyValueParser [private]

pointer to KeyValueParser, used for parsing key=value pairs

6.62.4.26 Logger* Langmuir::World::m_logger [private]

pointer to Logger, used for output

6.62.4.27 int Langmuir::World::m_maxDefects [private]

max number of defects

6.62.4.28 int Langmuir::World::m_maxElectrons [private]

max number of electrons

6.62.4.29 int Langmuir::World::m_maxHoles [private]

max number of holes

6.62.4.30 int Langmuir::World::m_maxTraps [private]

max number of traps

6.62.4.31 OpenCLHelper* Langmuir::World::m_ocl [private]

pointer to OpenCLHelper, used for Graphics Card calculations

6.62.4.32 SimulationParameters* Langmuir::World::m_parameters [private]

pointer to SimulationParameters

6.62.4.33 Potential* Langmuir::World::m_potential [private]

pointer to Potential, used for calculating the potential

6.62.4.34 boost::multi_array<double,3> Langmuir::World::m_R1 [private]

array of precomputed r values

This array is indexed by dx, dy, dz values, and r is in grid-units

6.62.4.35 boost::multi_array<double,3> Langmuir::World::m_R2 [private]

array of precomputed r-squared values

This array is indexed by dx, dy, dz values, and r is in grid-units

6.62.4.36 Random* Langmuir::World::m_rand [private]

pointer to Random, used for generating random numbers

6.62.4.37 RecombinationAgent* Langmuir::World::m_recombinationAgent [private]

pointer to electron/hole RecombinationAgent, used for removing Excitons

6.62.4.38 boost::multi_array<double, 3> Langmuir::World::m_sI [private]

self interaction, which is 1/(4 pi e e0 r), with r=1 grid unit When a charge at it's future site interacts with other charges at their current site, the charge will interact with it's own current site. So, this value needs to be subtracted off.

6.62.4.39 QList<SourceAgent*> Langmuir::World::m_sources [private]

list of SourceAgents

6.62.4.40 QList<int> Langmuir::World::m_trapSiteIDs [private]

list of trap sites

6.62.4.41 QList<double> Langmuir::World::m_trapSitePotentials [private]

list of trap potentials

6.62.4.42 QList<DrainAgent*> Langmuir::World::m_xDrains [private]

list of ExcitonSourceAgents

6.62.4.43 QList<SourceAgent*> Langmuir::World::m_xSources [private]

list of ExcitonSourceAgents

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

- /home/adam/opt/langmuir/src/langmuirCore/include/world.h

6.63 Langmuir::XYZWriter Class Reference

A class to output xyz files.

`#include <writer.h>`

Public Member Functions

- XYZWriter (World &world, const QString &name, QObject *parent=0)
  constructs the writer, has the same parameters as OutputInfo
- void write ()
  Write XYZ of the current step to the stream.

Protected Member Functions

- void writeVMDInitFile ()
  write a VMD script useful for opening the XYZ file

Protected Attributes

- World & m_world
  reference to the world object
- OutputStream m_stream
  output file stream

6.63.1 Detailed Description

A class to output xyz files.

6.63.2 Constructor & Destructor Documentation

6.63.2.1 Langmuir::XYZWriter::XYZWriter (World & *world,* const QString & *name,* QObject * *parent* = `0`)

constructs the writer, has the same parameters as OutputInfo

6.63.3 Member Function Documentation

6.63.3.1 void Langmuir::XYZWriter::write ( )

Write XYZ of the current step to the stream.

6.63.3.2 void Langmuir::XYZWriter::writeVMDInitFile ( ) [protected]

write a VMD script useful for opening the XYZ file

6.63.4 Member Data Documentation

6.63.4.1 OutputStream Langmuir::XYZWriter::m_stream [protected]

output file stream

6.63.4.2 World& Langmuir::XYZWriter::m_world [protected]

reference to the world object

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

- /home/adam/opt/langmuir/src/langmuirCore/include/writer.h

Chapter 7

File Documentation

7.1 /home/adam/opt/langmuir/src/langmuirCore/include/agent.h File Reference

```
#include <QTextStream>
#include <QMetaObject>
#include <QMetaEnum>
#include <QVector>
#include <QObject>
#include <QString>
#include <QDebug>
```

Classes

- class Langmuir::Agent

  A class that abstractly represents an object that can occupy grid sites.

Namespaces

- Langmuir

Functions

- QTextStream & Langmuir::operator<< (QTextStream &stream, const Agent::Type e)

  Output Agent type enum to stream.
- QDebug Langmuir::operator<< (QDebug dbg, const Agent::Type e)

  Output Agent type enum to debug information.

7.2 /home/adam/opt/langmuir/src/langmuirCore/include/chargeagent.h File Reference

```
#include "agent.h"
```

Classes

- class Langmuir::ChargeAgent

A class to represent moving charged particles.

- class Langmuir::ElectronAgent

  A class to represent moving negative charges.

- class Langmuir::HoleAgent

  A class to represent moving positive charges.

Namespaces

- Langmuir

7.3 /home/adam/opt/langmuir/src/langmuirCore/include/checkpointer.h File Reference

```
#include <QObject>
#include <QMap>
#include "parameters.h"
```

Classes

- class Langmuir::CheckPointer

  A class to read and write checkpoint files.

Namespaces

- Langmuir

Functions

- static std::ostream & Langmuir::operator<< (std::ostream &stream, QString &string)
- static std::istream & Langmuir::operator>> (std::istream &stream, QString &string)

7.4 /home/adam/opt/langmuir/src/langmuirCore/include/clparser.h File Reference

```
#include <QStringList>
#include <QString>
#include <QObject>
#include <QDebug>
#include <QMap>
```

Classes

- class Langmuir::CommandLineParser

  A class to parse command line arguments.

Namespaces

- Langmuir

7.5 /home/adam/opt/langmuir/src/langmuirCore/include/cubicgrid.h File Reference

```
#include "agent.h"
#include <QTextStream>
#include <QVector>
#include <QString>
#include <QObject>
#include <QDebug>
```

Classes

- class Langmuir::Grid

  A class to hold Agents, calculate their positions, and store the background potential.

Namespaces

- Langmuir

Functions

- QTextStream & Langmuir::operator<< (QTextStream &stream, const Grid::CubeFace e)

  Overload QTextStream for the Grid::CubeFace Enum.
- QDebug Langmuir::operator<< (QDebug dbg, const Grid::CubeFace e)

  Overload QDebug for the Grid::CubeFace Enum.

7.6 /home/adam/opt/langmuir/src/langmuirCore/include/drainagent.h File Reference

```
#include "fluxagent.h"
```

Classes

- class Langmuir::DrainAgent

  A class to remove charges.
- class Langmuir::ElectronDrainAgent

  A class to remove ElectronAgents.
- class Langmuir::HoleDrainAgent

  A class to remove HoleAgents.
- class Langmuir::RecombinationAgent

  A class to remove Excitons.

Namespaces

- Langmuir

7.7 /home/adam/opt/langmuir/src/langmuirCore/include/fluxagent.h File Reference

```
#include "agent.h"
#include "cubicgrid.h"
```

Classes

- class Langmuir::FluxAgent

  A class to change the number of carriers in the system.

Namespaces

- Langmuir

7.8 /home/adam/opt/langmuir/src/langmuirCore/include/gzipper.h File Reference

```
#include <QString>
```

Functions

- QString gunzip (QString fileName, bool *wasZipped=NULL)

  gunzip a file using QProcess
- QString gzip (QString fileName)

  gzip a file using QProcess

7.8.1 Function Documentation

7.8.1.1 QString gunzip (QString *fileName*, bool * *wasZipped* = NULL)

gunzip a file using QProcess

Parameters

fileName	name of file to gunzip
msecs	timeout time

Returns

altered file name

7.8.1.2 QString gzip (QString *fileName*)

gzip a file using QProcess

Parameters

fileName	name of file to gzip
msecs	timeout time

Returns

altered file name

7.9 /home/adam/opt/langmuir/src/langmuirCore/include/keyvalueparser.h File Reference

```
#include <QObject>
#include "variable.h"
#include "parameters.h"
#include "world.h"
```

Classes

- class Langmuir::KeyValueParser

  A class to read the parameters and store them in the correct place.

Namespaces

- Langmuir

7.10 /home/adam/opt/langmuir/src/langmuirCore/include/nodefileparser.h File Reference

```
#include <QStringList>
#include <QVector>
#include <QString>
#include <QObject>
#include <QDebug>
#include <QList>
#include <QMap>
```

Classes

- class Langmuir::NodeFileParser

Namespaces

- Langmuir

7.11 /home/adam/opt/langmuir/src/langmuirCore/include/openclhelper.h File Reference

```
#include <QObject>
#include <QVector>
```

Classes

- class Langmuir::OpenClHelper

  A Class to run OpenCL calculations.

Namespaces

- Langmuir

Macros

- #define __CL_ENABLE_EXCEPTIONS

7.11.1 Macro Definition Documentation

7.11.1.1 #define __CL_ENABLE_EXCEPTIONS

7.12 /home/adam/opt/langmuir/src/langmuirCore/include/output.h File Reference

```
#include "parameters.h"
#include <QTextStream>
#include <QObject>
#include <QFile>
```

Classes

- class Langmuir::OutputInfo

  A class to generate file names using the SimulationParameters.
- class Langmuir::OutputStream

  A class to combine QFile, QTextStream and OutputInfo (QFileInfo).

Namespaces

- Langmuir

Functions

- QTextStream & newline (QTextStream &s)

  put a newline character in the stream that ignores the streams current FieldWidth
- QTextStream & space (QTextStream &s)

  put a space in the stream that ignores the streams current FieldWidth
- void Langmuir::backupFile (const QString &name)

  Back up a file.

7.12.1 Function Documentation

7.12.1.1 QTextStream& newline (QTextStream & *s*)

put a newline character in the stream that ignores the streams current FieldWidth

7.12.1.2 QTextStream& space (QTextStream & *s*)

put a space in the stream that ignores the streams current FieldWidth

7.13 /home/adam/opt/langmuir/src/langmuirCore/include/parameters.h File Reference

```
#include <QDateTime>
#include <QFileInfo>
#include <QDebug>
#include <cmath>
#include <QDir>
```

Classes

- struct Langmuir::ConfigurationInfo

  A struct to temporarily store site IDs.
- struct Langmuir::SimulationParameters

  A struct to store all simulation options To add new variables, follow these steps:

Namespaces

- Langmuir

Functions

- void Langmuir::setCalculatedValues (SimulationParameters &par)

  sets parameters that depend upon other parameters
- void Langmuir::checkSimulationParameters (SimulationParameters &par)

  check the parameters, making sure they are valid

7.14 /home/adam/opt/langmuir/src/langmuirCore/include/potential.h File Reference

```
#include <QObject>
#include "boost/multi_array.hpp"
```

Classes

- class Langmuir::Potential

  A class to calculate the potential.

Namespaces

- Langmuir

Macros

- #define BOOST_DISABLE_ASSERTS

7.14.1 Macro Definition Documentation

7.14.1.1 #define BOOST_DISABLE_ASSERTS

7.15 /home/adam/opt/langmuir/src/langmuirCore/include/rand.h File Reference

```
#include <QObject>
#include <QDataStream>
#include <QTextStream>
#include <boost/random.hpp>
#include <ctime>
```

Classes

- class Langmuir::Random

  A class to generate random numbers.

Namespaces

- Langmuir

7.16 /home/adam/opt/langmuir/src/langmuirCore/include/simulation.h File Reference

```
#include <QObject>
```

Classes

- class Langmuir::Simulation

  A class to orchestrate the calculation.

Namespaces

- Langmuir

7.17 /home/adam/opt/langmuir/src/langmuirCore/include/sourceagent.h File Reference

```
#include "fluxagent.h"
```

Classes

- class Langmuir::SourceAgent

  A class to inject charges.
- class Langmuir::ElectronSourceAgent

  A class to inject ElectronAgents.
- class Langmuir::HoleSourceAgent

  A class to inject HoleAgents.

- class Langmuir::ExcitonSourceAgent

  A class to inject Excitons.

Namespaces

- Langmuir

7.18 /home/adam/opt/langmuir/src/langmuirCore/include/variable.h File Reference

```
#include <QTextStream>
#include <QDateTime>
#include <QObject>
#include <QDebug>
#include <limits>
#include <ostream>
```

Classes

- class Langmuir::Variable

  A class to map between variable names (keys) and locations (references)
- class Langmuir::TypedVariable< T >

  A template class to map between variable names (keys) and locations (references)

Namespaces

- Langmuir

Functions

- QTextStream & Langmuir::operator<< (QTextStream &stream, const QDateTime &datetime)

  output QDateTime as qint64 mSecsSinceEpoch
- QTextStream & Langmuir::operator<< (QTextStream &stream, const Variable &variable)

  overload operator to write keyValue() to a stream
- QDebug Langmuir::operator<< (QDebug dbg, const Variable &variable)

  overload operator to write keyValue() to a QDebug
- std::ostream & Langmuir::operator<< (std::ostream &stream, Variable &variable)

  Operator overload to output to output 'key = value' to std::ostream.

7.19 /home/adam/opt/langmuir/src/langmuirCore/include/world.h File Reference

```
#include <QtCore>
#include <QtGui>
#include "boost/multi_array.hpp"
```

Classes

- class Langmuir::World

  A class to hold all objects in a simulation.

Namespaces

- Langmuir

Macros

- #define BOOST_DISABLE_ASSERTS

7.19.1 Macro Definition Documentation

7.19.1.1 #define BOOST_DISABLE_ASSERTS

7.20 /home/adam/opt/langmuir/src/langmuirCore/include/writer.h File Reference

```
#include <QObject>
#include <QPainter>
#include <QColor>
#include <QImage>
#include "output.h"
```

Classes

- class Langmuir::XYZWriter

  A class to output xyz files.
- class Langmuir::FluxWriter

  A class to output source and drain info.
- class Langmuir::CarrierWriter

  A class to output carrier stats (lifetime and pathlength)
- class Langmuir::ExcitonWriter

  A class to output exciton stats (lifetime and pathlength)
- class Langmuir::GridImage

  A class to draw images of the grid.
- class Langmuir::Logger

  A class that organizes output.

Namespaces

- Langmuir

7.21 /home/adam/opt/langmuir/src/langmuirView/include/axis.h File Reference

```
#include "sceneobject.h"
```

Classes

- class Axis

  A class to represent an xyz axis.

7.22 /home/adam/opt/langmuir/src/langmuirView/include/box.h File Reference

```
#include "sceneobject.h"
#include <QOpenGLBuffer>
```

Classes

- class Box

  A class to represent a textured box.

7.23 /home/adam/opt/langmuir/src/langmuirView/include/color.h File Reference

```
#include <QObject>
#include <QColor>
```

Namespaces

- color

Functions

- float * color::qColorToArray4 (const QColor &color, float *array)

  Copy color data to array of size 4.
- float * color::qColorToArray4 (const QColor &color)

  Copy color data to array of size 4 (static)

7.24 /home/adam/opt/langmuir/src/langmuirView/include/colorbutton.h File Reference

```
#include <QColorDialog>
#include <QPushButton>
#include <QColor>
#include <QDebug>
```

Classes

- class ColorButton

7.25 /home/adam/opt/langmuir/src/langmuirView/include/corneraxis.h File Reference

```
#include "axis.h"
```

Classes

- class CornerAxis

  A class to represent an xyz axis that doesnt change size/position.

7.26 /home/adam/opt/langmuir/src/langmuirView/include/grid.h File Reference

```
#include "sceneobject.h"
#include <QOpenGLShaderProgram>
#include <QOpenGLShader>
#include <QOpenGLBuffer>
#include <QMatrix4x4>
#include <QVector>
```

Classes

- class Grid

  A class to represent simulation grid.

7.27 /home/adam/opt/langmuir/src/langmuirView/include/gridview.h File Reference

```
#include "chargeagent.h"
#include "parameters.h"
#include "simulation.h"
#include "cubicgrid.h"
#include "world.h"
#include <QtCore>
#include <QtGui>
#include <QGLShaderProgram>
#include <QMatrix4x4>
#include <QGLWidget>
#include <QGLBuffer>
#include <QDialogButtonBox>
#include <QDesktopWidget>
#include <QApplication>
#include <QInputDialog>
#include <QDockWidget>
#include <QMessageBox>
#include <QFileDialog>
#include <QPushButton>
#include <QMainWindow>
#include <QStatusBar>
#include <QDoubleSpinBox>
#include <QColorDialog>
#include <QGridLayout>
#include <QLCDNumber>
#include <QCheckBox>
#include <QComboBox>
#include <QLabel>
```

Classes

- class Langmuir::ColoredObject
- class Langmuir::Box
- class Langmuir::PointArray
- class Langmuir::DSpinBox
- class Langmuir::SSpinBox

- class Langmuir::CheckBox
- class Langmuir::Button
- class Langmuir::RecordDialog
- class Langmuir::GridViewGL
- class Langmuir::Navigator
- class Langmuir::SceneOptions
- class Langmuir::Controls
- class Langmuir::MainWindow

Namespaces

- Langmuir

7.28 /home/adam/opt/langmuir/src/langmuirView/include/isosurface.h File Reference

```
#include <QVector3D>
#include <QVector>
#include <QObject>
#include <QList>
#include "boost/multi_array.hpp"
```

Classes

- class MarchingCubes::Triangle

  Container for vertices and normals of triangle.
- class MarchingCubes::Isosurface

  A class to compute a contour iso-surface.

Namespaces

- MarchingCubes

Typedefs

- typedef boost::multi_array
  < float, 3 > MarchingCubes::scalar_field

Variables

- static const float MarchingCubes::a2fVertexOffset [8][3]
- static const int MarchingCubes::a2iEdgeConnection [12][2]
- static const float MarchingCubes::a2fEdgeDirection [12][3]
- static const int MarchingCubes::aiCubeEdgeFlags [256]
- static const int MarchingCubes::a2iTriangleConnectionTable [256][16]

7.29 /home/adam/opt/langmuir/src/langmuirView/include/isosurfacedialog.h File Reference

```
#include "mesh.h"
#include <QDialog>
```

Classes

- class IsoSurfaceDialog

Namespaces

- Ui

7.30 /home/adam/opt/langmuir/src/langmuirView/include/langmuirviewer.h File Reference

```
#include <QGLViewer/qglviewer.h>
#include <QGLViewer/manipulatedCameraFrame.h>
#include <QErrorMessage>
#include <QColorDialog>
#include <QMatrix4x4>
#include <QSettings>
#include "isosurface.h"
#include "corneraxis.h"
#include "pointcloud.h"
#include "light.h"
#include "grid.h"
#include "rand.h"
#include "mesh.h"
#include "box.h"
#include "boost/multi_array.hpp"
```

Classes

- class LangmuirViewer

  Widget to view Langmuir Simulation in real time.

Namespaces

- Langmuir

Macros

- #define BOOST_DISABLE_ASSERTS

7.30.1 Macro Definition Documentation

7.30.1.1 #define BOOST_DISABLE_ASSERTS

7.31 /home/adam/opt/langmuir/src/langmuirView/include/light.h File Reference

```
#include "sceneobject.h"
#include <QVector4D>
```

Classes

- class Light

  A class to represent a light source.

7.32 /home/adam/opt/langmuir/src/langmuirView/include/mainwindow.h File Reference

```
#include <QCloseEvent>
#include <QMainWindow>
#include <QAction>
#include <QStyle>
#include <QIcon>
#include <QDir>
```

Classes

- class MainWindow

  A window with an OpenGL widget.

Namespaces

- Ui

7.33 /home/adam/opt/langmuir/src/langmuirView/include/mesh.h File Reference

```
#include "sceneobject.h"
#include <QOpenGLShaderProgram>
#include <QOpenGLShader>
#include <QOpenGLBuffer>
#include <QMatrix4x4>
#include <QVector>
```

Classes

- class Mesh

  A class to represent a mesh.

Functions

- Q_DECLARE_METATYPE (Mesh::Mode)

7.33.1 Function Documentation

7.33.1.1 Q_DECLARE_METATYPE (Mesh::Mode)

7.34 /home/adam/opt/langmuir/src/langmuirView/include/pointcloud.h File Reference

```
#include "sceneobject.h"
#include <QOpenGLShaderProgram>
#include <QOpenGLShader>
#include <QOpenGLBuffer>
#include <QMatrix4x4>
#include <QVector>
```

Classes

- class PointCloud

  A class to represent a point cloud.

Functions

- Q_DECLARE_METATYPE (PointCloud::Mode)

7.34.1 Function Documentation

7.34.1.1 Q_DECLARE_METATYPE (PointCloud::Mode)

7.35 /home/adam/opt/langmuir/src/langmuirView/include/pointdialog.h File Reference

```
#include "pointcloud.h"
#include <QDialog>
```

Classes

- class PointDialog

Namespaces

- Ui

7.36 /home/adam/opt/langmuir/src/langmuirView/include/sceneobject.h File Reference

```
#include <QVector>
#include <QObject>
#include <QColor>
#include <QDebug>
#include <GL/glu.h>
```

Classes

- class SceneObject

  Base class for objects in OpenGL scene.

Index